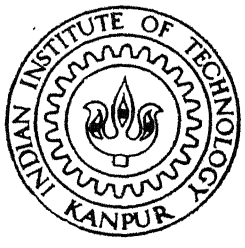
On Design of Multicast ATM Switches

by

JAYESH V. SHAH



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

FEBRUARY 1997

On Design of Multicast ATM Switches

A Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

Master of Technology

by

Jayesh V. Shah

to the

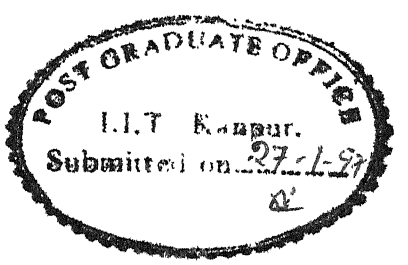
DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January 1997

EE-1997-M-SHA-ON

Certificate

This is to certify that the work contained in the thesis entitled **On Design of Multicast ATM Switches** by **Jayesh V. Shah** has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.



Dr. D. Manjunath,
Assistant Professor,
Department of Electrical Engineering,
Indian Institute of Technology, Kanpur.

Dr. Subir. K. Roy,
Assistant Professor,
Department of Electrical Engineering,
Indian Institute of Technology, Kanpur.

Acknowledgements

First of all, I would like to express my sincere gratitude to my thesis supervisors Dr. D. Manjunath and Dr. Subir K. Roy for their invaluable guidance and constant encouragement throughout the work. I would like to thank them for giving me complete freedom in work. I am very thankful to them for giving me an opportunity to undertake this challenging topic as my M. Tech. thesis. It was their wonderful association, that enabled me to achieve the goal of this work.

I would like to thank Dr. Dheeraj Sanghi for taking me deep into the field of computer networks. I would also like to thank Dr. S. K. Bose for introducing me to the concepts of queueing theory. Special thanks to Vipul, Utpal, Deepak and Pankaj for their help and constant encouragement. I thank all my friends, especially those belonging to the Telecom group, for their memorable company during my stay at IITK. I would also like to thank all the members of the Telematics group especially Mr. Navpreet Singh and Mrs. Sandhya Sule for their constant help and guidance.

No words can be said or written to express my feelings towards my family members. I am very thankful for their constant support and encouragement.

Last, but not the least, I would like to thank all the IITK community for providing me with a peaceful ambience and a home away from home.

Jayesh V. Shah

Abstract

Asynchronous Transfer Mode (ATM) network technology is widely acknowledged as a key component of the emerging global information infrastructure. ATM is basically a connection oriented, packet switched technology, with very small packets called cells, used to transfer the information. However, to be useful in large networks, ATM technology requires a highly efficient switching system that allows networks that are more cost effective than is possible with most traditional designs.

Broadband Integrated Services Digital Networks (B-ISDN) based on ATM are expected to support new services like videoconferencing, video-on-demand, multiparty telephony, distributed computing, video distribution and broadcasting and tele-teaching which are inherently multicast in nature. To support such services multicast service should be provided by the underlying network in addition to the usual unicast service.

The present thesis is devoted to providing multicast support in ATM switches. We propose a new architecture for supporting multicast in the ATM switches. Apart from the drastic reductions in the memory required for storing the connection control information, the proposed architecture achieves better throughputs than that achievable using the existing architectures. In the later part of the thesis, we suggest a method to construct large scalable multicast ATM switches and propose an architecture for its realization. We conclude the thesis with an example illustrating the design of a 1024×1024 switch.

Contents

List of Acronyms

A	Activation Bit
A-Bus	Address Bus
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Services Digital Network
BBN	Broadcast Banyan Network
BE	Buffer Empty
BGT	Broadcast and Group Translator
BST	Buffer-Scheduler-Translator
CAM	Content Addressable Memory
CCT	Connection Control Table
CLP	Cell Loss Priority
CN	Average number of Copies
CRC	Cyclic Redundancy Check
CRP	Collision Resolution Processor
CT	Copy Table
DAE	Dummy Address Encoder
ES	Exit Stamp
FB	FeedBack signal
FBU	FeedBack Unit
FIFO	First In First Out
GS-AE	Group Splitter and Address Encoder

HOL	Head Of the Line
HOL-FO	Head Of the Line FanOut
I-Bus	Index Bus
IAQ	Idle Address Queue
ICP	Input Cell Processor
LMS	Lee Multicast Switch
MO-LMS	Memory Optimized-Lee Multicast Switch
P-Bus	Priority Bus
PSC	Parallel to Serial Convertor
QoS	Quality of Service
RAM	Random Access Memory
RAN	Running Adder Network
RAQ	Read Address Queue
RBN	Replicating Banyan Network
RH	Routing Header
RN	Routing Network
RS	Running Sum
RT	Routing Table
SCOQ	Shared Concentration and Output Queueing
SDGN	Space Division Grouping Network
SMXQ	Sharing with MaXimum Queue constraint
SPC	Serial to Parallel Convertor
TT	Translation Table
max	maximum address
min	minimum address
VC	Virtual Circuit
VCI	Virtual Circuit Identifier
VLSI	Very Large Scale Integration
VPI	Virtual Path Identifier
WT	Waiting Time

List of Figures

Chapter 1

Introduction

1.1 ATM Switching System

Asynchronous Transfer Mode (ATM) network technology is widely acknowledged as a key component of the emerging global information infrastructure. It facilitates the integration of a wide variety of applications such as classical data applications, image-retrieval and real time audio and video, on a common switching and transmission platform. ATM is basically a connection oriented, packet switched technology, with very small packets called cells, used to transfer the information. The transmission technology is asynchronous with demand based slot assignment. However, to be useful in large networks, ATM technology requires a highly efficient switching system that allows networks that are more cost effective than is possible with most traditional designs.

An ATM switching system does much more than simply transport of cells from an input port to appropriate output ports. In addition to routing, the switching system has to perform a variety of call processing and control functions for each connection like admission control, network routing, cell scheduling, buffer management, flow control, congestion control, switch management and signaling. All these call processing and network support functions have to be embedded in the switch to provide effective ATM support. A three layered architecture for the ATM switching system was proposed in [14]. A modified figure of the architecture showing more internal details is shown in Figure 1.1.

The functions like admission control, signaling and routing are a part of the Call

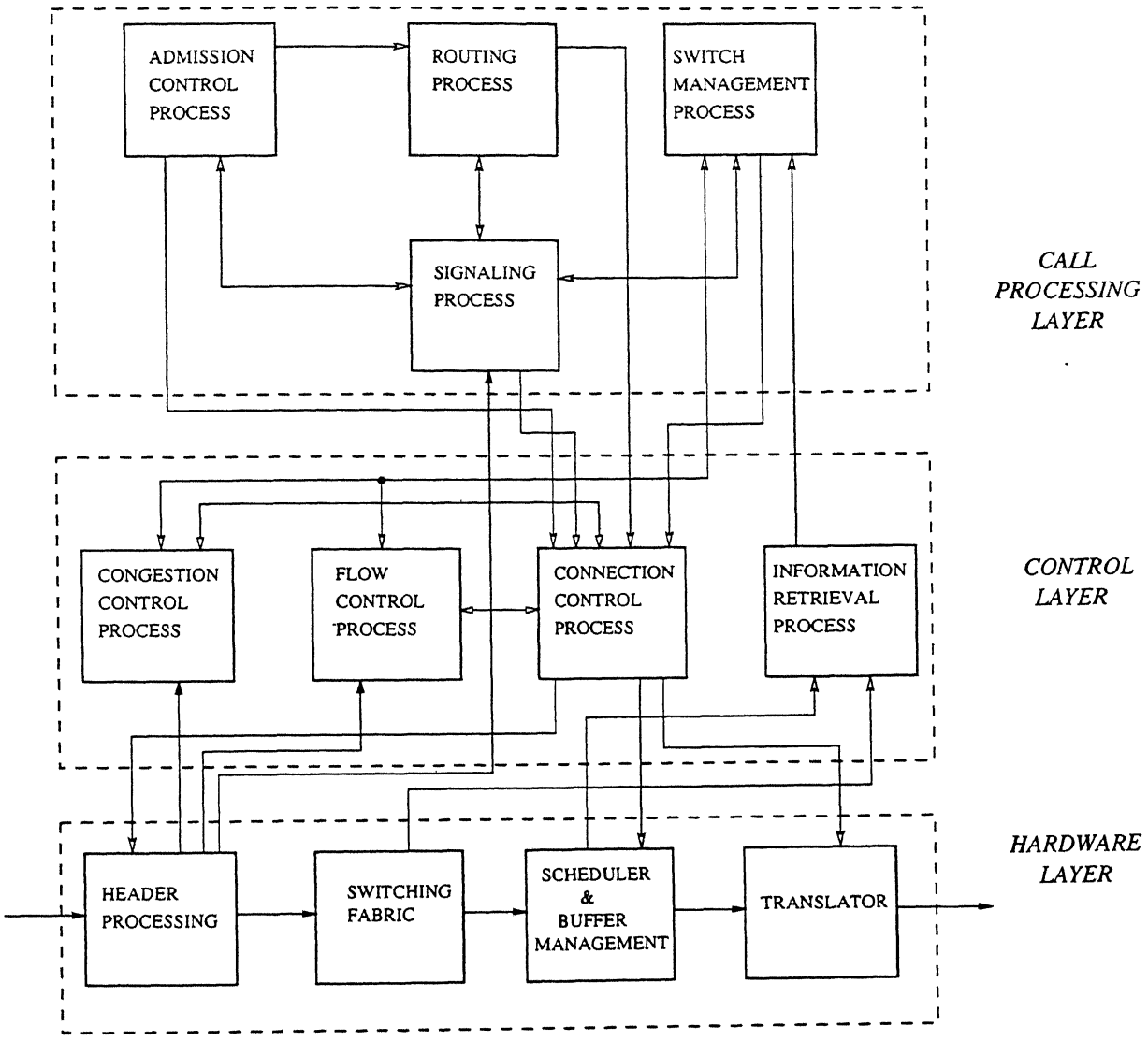


Figure 1.1: Layered Architecture for an ATM Switching System

processing layer and are activated on a per connection basis The admission control process determines whether the acceptance of the new call will affect the existing connections in terms of their acceptable QoS parameters If the new connection is acceptable, the connection request is forwarded to the routing process which determines the next node in the sequence The signaling process facilitates the internode communication Apart from the admission control and routing pocesses, a switch management process is needed to provide an interface to the network management and interface processes Most of the functions of this layer are invoked on a per connection basis and hence are implemented in software

The **Hardware layer** gets valid cells from the physical layer after cell delineation and header CRC check The ATM layer processing is done in the hardware layer It contains the switch fabric, responsible for the routing of the cells In order to allow the upper layers to access the status information, it provides suitable interfaces to the upper layers To support the emerging demand for the multicast traffic it must also have the multicast function built into it The hardware layer also contains the connection control tables which, apart from governing the functioning of the hardware layer, also provide a control interface to the upper layer To avoid the cell loss due to output contention and internal blocking in the switch fabric, adequate buffering is provided in this layer To support the diverse QoS requirements and for effective congestion and flow control, it incorporates bandwidth and buffer management strategies embedded in the scheduler It can be easily seen that all these functions need to be invoked on a per cell basis and have real time constraints Hence they are implemented in hardware

The **Control layer** acts as a bridge between the two layers discussed above It contains processes that are invoked more frequently than those in the call processing layer to control the functioning of the hardware layer It provides an interface between the call processing layer and the hardware layer called the Information Retrieval Process facilitating the retrieval of the status information by the call processing layer from the hardware layer This layer provides a connection control interface to allow the call processing layer processes to control the functioning of the hardware layer by modifying the contents of the Connection Control Tables (CCTs) which govern the functioning of the hardware layer Apart from these processes meant for the interface between the top and

the bottom layer, the congestion control process and the flow control processes also reside in this layer The frequency of activation of the processes in this layer lie in between those of the call processing and the hardware layers and hence the functions of this layer can be implemented either in hardware or in software

1.2 ATM Switch Fabrics

From the discussion above, it is clear that the design of the hardware layer, because of its real time processing requirements is expected to dominate the hardware complexity of a switching system Our interest in this thesis is in improving the complexity of this hardware layer

Although the functionality required of the ATM switches is quite simple and is similar to that required of traditional store and forward packet switches, the design challenge here is to meet the speed requirements For example, a 155 Mbps line speed allows only 2 73 microseconds for the switch to read the header, decode the routing information in the form of Virtual Circuit Identifier and Virtual Path Identifier (VCI/VPI), route the cell to the appropriate output port and to translate the header of the outgoing cell The asynchronous nature of the transfer technology puts further constraints as each cell is individually switched using the routing information encoded in its header Therefore the effectiveness of a switch is strongly dependent on the underlying architecture and the implementation technology

Three basic types of packet switches are proposed in the literatue

- Shared memory switches
- Shared Medium switches
- Space division switches

Shared memory ATM switches consist of a single memory shared by all the input and output ports. All the incoming cells are written into this common memory Internally the cells are organized into separate output queues, one for each output port The output ports read the cells from their respective queues, one at a time

In this type of architecture, a number of design constraints limit the size of the switch These include the processing time of the central controller to manage the queues because it should be capable of processing sequentially N input cells and selecting N outgoing cells in each time slot, for an $N \times N$ switch The memory access time and the bandwidth required also put a limit, because the memory must be accessed N times per slot for enqueueing the cells and N times for dequeueing cells

In **Shared medium** ATM switches, all the cells arriving in a particular time slot are synchronously multiplexed onto a common high speed medium, typically a parallel bus of bandwidth N times the rate of individual input ports The output ports are connected to this bus through the address filters which filter out the cells that are not destined for the particular port The critical design constraints here are the speed of the bus and the buffers The bus and the associated address filters are required to operate at a rate N times that of the individual ports while the buffers at the output ports need to be operated at a rate (N+1) times that of the port

Space division ATM switches provide multiple concurrent paths from an input to corresponding outputs through the switch Each of these paths operate at port speed Thus the size of this kind of a switch is not limited by the speed of the memory elements Most of the popular space-division ATM architectures use *banyan networks*, consisting of small switch elements arranged in a multi stage interconnection network topology The routing through the switch fabric can be distributed, leading to a *self routing* switching network The banyan network possesses a regular structure which is particularly amenable to VLSI implementation Also, the modular structure allows construction of larger networks from smaller ones without the need for modification in the physical layout or in the algorithms needed for their operation Thus these networks are inherently scalable to larger sizes However, the throughput of these switches is limited by internal blocking and output contention

1.3 Multicast Support in ATM Switches

Broadband Integrated Services Digital Networks (B-ISDN) based on ATM technology are expected to support new services like videoconferencing, video-on-demand, multiparty

telephony, distributed computing, video distribution and broadcasting and tele-teaching which are inherently multicast in nature To support such services multicast service should be provided by the underlying network in addition to the usual unicast service Thus the fast packet switches in the ATM are required to have built in multicast capability to support such services

An ATM multicast switch is a generalization of the unicast switch in the sense that it needs to send a cell from an input port to a predetermined set of output ports. Thus, the multicast ATM switch needs to consider issues like cell replication, addressing and maintenance of connection control information for the copies in addition to handling the routing The challenge in the design of such a switch is to add multicast support in the switch without incurring a significant penalty on unicast connections while maintaining a minimum hardware complexity [14]

It is relatively easy to support multicasting in the shared memory and shared medium kind of switches For example, in shared memory switches, which store all the input cells in a common memory pool, multicasting can be easily achieved by repeated reading of the same cell by corresponding output ports and retaining the cell in the memory until all the copies have been read Thus, multicasting can be achieved at the cost of some additional complexity in the design of the memory controller

On the other hand, space division switches require additional dedicated support for multicasting The most popular approach uses a two stage process, the first of which generates the copies and the second is responsible for the routing of the copies to the output ports The presence of a number of copies of the same cell in the switching network also increases the blocking probability and the buffer requirements

In this thesis, we develop a new architecture for multicasting in space-division ATM switches This architecture offers huge savings in the memory required for connection control information It also uses shared buffering in an elegant manner to reduce the overflow

1.4 Large Multicast ATM Switches

Large switches have an inherent cost and performance advantage when it comes to building a large network A large number of users can be networked by properly interconnecting small switches with modest number of output ports However, as Turner et al [3] show that even most efficient interconnection topology leads to the aggregate number of ports totalled over all the switches much larger than the number of users For efficient switch architectures, the total system cost is dominate by the per port cost, making the cost of such a hierarchical network much larger than that if we could interconnect all the users using a single large switch This effect is termed as Small Switch Penalty in [3] In addition to have higher cost, the hierarchical network design yields greatly inferior performance with respect to virtual circuit blocking, queueing delay and cell loss Its blocking performance is particularly poor for multicast connections

As we have already seen, space division switches are inherently scalable in nature Their modular architecture and easy amenability to VLSI fabrication allows the basic architecture to be extended to very large number of ports As compared to them, the shared memory switches are limited in size because of the technological constraints regarding the memory size and speed and the complexity of the memory controller

The requirement of multicast functionality puts further hurdles in making large switches as the requirement of multicast and scalability conflict with each other

In order to solve this problem, in this thesis we propose a hybrid scheme combining the features of both the space-division and shared-memory switches to construct large multicast ATM switches

1.5 Organization of The Thesis

In this thesis, we propose a new scheme to support multicast in the space division ATM switches It brings about drastic reductions in terms of the memory required to maintain the connection control information, as compared to the existing proposals We also incorporate elaborate mechanisms to reduce the loss and unfairness in our architecture leading to superior performance characteristics In the later part of the thesis, we propose an architecture to build very large ATM switches, with multicast capability, by combining

the space-division and shared-memory approaches We conclude the thesis, with an example design of a 1024×1024 multicast ATM switch

The rest of the thesis is organized as follows Chapter 2 deals with providing efficient multicast support in the space division swithcing fabrics. This is followed by Chapter 3, in which we discuss the concept of channel grouping in ATM switches In Chapter 4, we give an overview of the shared memory switching techniques In Chapter 5, we combine the concepts developed in the previous chapters to propose a new architecture for building large ATM switches with built in multicast support Chapter 1 concludes the thesis and gives directions for future extension of our work

Chapter 2

Multicasting in Space-Division Switches

2.1 Introduction

As we have seen in Chapter 1, space division switches require elaborate mechanisms for efficient multicast support Depending on how the copies are made, two basic design paradigms have been proposed for providing multicast capability in a space division switch - *space based* and *time based* [26] In the space based multicast switch all the required copies of the input cell are produced simultaneously by a *Copy Network* that replicates the cells At the output of the copy network, the routing information is attached to the individual copies and the copies are fed to the *Routing Network* which routes the copies to their destination ports In the time based multicast switch, the cells are replicated in time rather than in space The original cell is buffered and the same buffer is read repeatedly, once for each copy The list of the destinations is maintained as a linked list and the copies are produced sequentially This approach saves on hardware that is required for replicating of cells and hence reduces complexity of hardware However, it introduces a delay corresponding to the copying process and could lead to congestion if the fanout of the arriving cells is large [26] This approach is not preferred for real time multicast ATM switches because although the delay can be reduced by a speedup of the buffer readouts, the speedup is expensive

Lea suggests a multicast switch architecture that combines the two paradigms [26] In

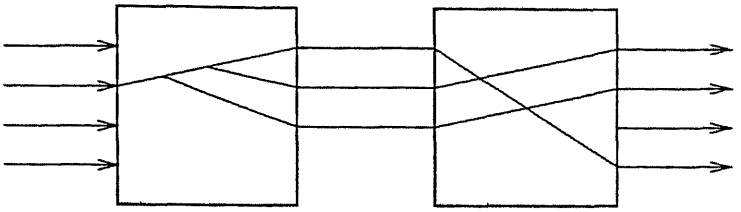


Figure 2 1 Space-Based Multicast Switch

this architecture, the list of all the destinations of a particular input cell is divided into a number of smaller lists called segments The segments, apart from the destinations, may also contain entries referencing to other segments, which may be processed by different input port processors, to achieve load balancing Turner also suggests a similar scheme in which the original cell is used to produce secondary cells, which can be handled by different input port processors in parallel to generate more copies [45]. For example, to make F copies of a cell, it takes $\log_2 F$ time slots

In this chapter our interest is in space based multicast switches that use a copy network to achieve cell replication A multicast ATM switch using the space based method consists of a serial combination of a copy network and a point to point switch [27] (See Figure 2 1) The copy network makes copies of the input cells as required by the connection They are then routed to the appropriate output port by the routing network The routing network is essentially a point to point switch

There are many proposals of space based multicast switches in literature In [43], Turner proposes an architecture in which the copies are produced by a copy network that uses the delayed copying algorithm The copy network consists of $\log_2 N$ stages of $\frac{N}{2}$ elements However if a cell needs F copies, the copying process will be done in the final $\lceil \log_2 F \rceil$ stages A disadvantage of this copy network is that it is blocking in nature and requires internal buffering to reduce blocking Further, supporting dynamic group changes is difficult in this switch because complicated table management procedures are required to support such changes [26]

Liew [29, 30] proposes a copy network architecture using a shuffle network In order to reduce the blocking probability, he suggests use of large number of stages in the shuffle network The architecture, although simple, is inherently blocking in nature and also

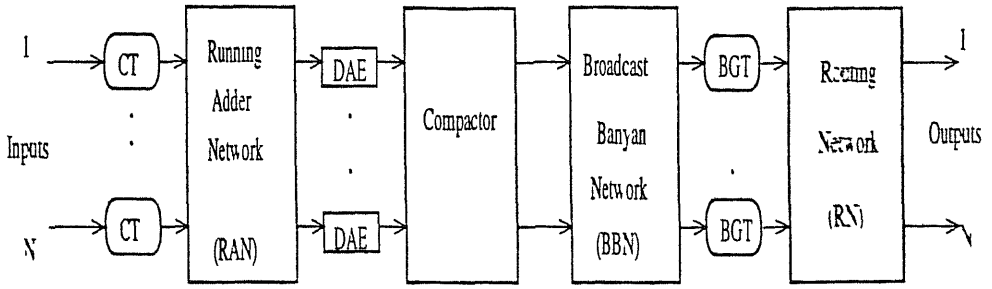


Figure 2 2 Block Diagram of the Lee Multicast Switch (LMS)

needs special protection to avoid deadlock situations

The multicast ATM switch proposed by Lee [27] is the most well known architecture (See Figure 2 2) A non-blocking, self-routing, constant latency copy network is used to make the copies It proposes a broadcast banyan network with 2×2 switching elements interconnected in the banyan topology The switching elements have an additional capability of duplicating the cells to both the output ports Lee's copy network replicates packets through an encoding and decoding process In the encoding process, a running adder network and dummy address encoders transform the set of copies to be produced into a set of monotonic address intervals, which are inserted into the packet headers After passing through the concentrator, which compacts the active inputs, these packets enter the broadcast banyan network This preprocessing of the cells before feeding them to the broadcast banyan network is essential to make the broadcast banyan network nonblocking The decoding process uses the *boolean interval splitting algorithm* for the broadcast banyan network, with each BBN switching element using a two-bit header information to make the copies of the cell Two different tables are required in the lee multicast switch, *Copy Table (CT)* at the input to supply the information about the number of copies wanted for a cell and *Broadcast and Group Translator (BGT)* at the output to supply the control information like priorities, outgoing Virtual Circuit Identifier/Virtual Path Identifier (VCI/VPI) and output port number, to the individual copies

We believe that apart from some limitations, explained in the following sections, the Lee Multicast Switch (LMS) using the concept of self routing, constant latency, nonblocking copy network is one of the best architectures for multicasting in space-division ATM switches Its simple repetitive structure is easily amenable for VLSI implementation Moreover, the transit delay through the network is very small

2.2 Table Duplication in Lee Multicast Switch

In [14] Gupta showed that the complexity of an ATM switch will be governed primarily by the memory elements used inside the switch for the tables that store the control information associated with the virtual circuits (VCs) such as priorities, outgoing VCI/VPI and output port number

In the LMS, the copying process is such that the copies of a cell from an input port can appear at any output port of the copy network depending on the traffic on other input ports and their fanouts Hence, the routing information about *all* the copies of *all* the connections passing through the switch have to be maintained in the *BGTs* at each of the outputs of the copy network This amounts to a large memory requirement even for very small switches For example, as will be shown in section 2 3 1, a 256×256 switch, supporting 3000 VCs per port, will require 36 9 Mbits of memory per port, assuming a control information (excluding the output port information) of 40 bits per output VC This large storage requirement is primarily due to the duplication of *BGT* entries and we will call this the *table duplication problem*

A large number of tables also complicates the table management procedure A single addition or drop from a group needs changing the entries in a number of tables

A solution to the table duplication problem and consequent reductions in the memory requirement will make this architecture very attractive In the following we first discuss some of the modifications suggested in literature to the Lee Multicast Switch to reduce the effect of the table duplication problem

Turner suggests some solutions to the table duplication problem [44] The first one is to share a single *BGT* among a number of output lines This multiplexing reduces the storage requirement for the *BGTs* by requiring us to maintain fewer of them To multiplex a *BGT*, either the cells have to be delayed or access to the *BGTs* have to be speeded up The multiplexing also substantially increases the overhead due to additional functions such as managing sharing of *BGTs*

In [14], Gupta suggests a solution to reduce the effect of table duplication problem by utilizing the fact that the unicast traffic does not require the services of the copy network Thus, the unicast cells can bypass the copy network and be submitted to the routing network This will require a split in the BGTs (Gupta calls them connection

control tables (CCTs)) to separate the information for the unicast and the multicast traffic Since the information needed for the unicast connections need not be duplicated, the storage requirement for the *CCTs* is decreased Although this scheme achieves some reductions, the amount of reductions are highly dependent on the relative values of the upper limits on simultaneous unicast and multicast connections through a port Inspite of these reductions in the memory requirements, the problem of the duplication of the *CCT* information persists and dominates the memory size when the number of allowable simultaneous multicast connections is large

Other modifications propose changing the copying algorithm in the copy network to force the cell copies to appear only at a set of predetermined outputs [45, 40] These techniques introduce blocking in the copy network and are of limited value

A modification by Liew [29], suggests use of a parallel copy network along with the one carrying the cells, dedicated for the transmission of table entries However, this approach results in total duplication of the hardware which may not be acceptable

An interesting proposal by Min et al [33], modifies the *BGT* entries in order to decrease the memory size per *BGT* It suggests the use of a global virtual table (GVT) (Figure 2 2) in which each row corresponds to one input virtual channel and each column contains the control information corresponding to an output VC fanning out of this input VC The number of columns equals the number of output ports Each table entry is of constant size and each row is completed by repeating the entries as many times as necessary Since the copies from a single input cell exit contiguously from the outputs of the copy network, the ith column of the GVT contains, for each multicast connection, the proper output address that the ith *BGT* needs to provide Thus, the *BGTs* don't need to keep the routing information for all the copies of a particular cell and the information for only one copy suffices If an input cell has a fanout of F, any consecutive F columns of the GVT contain the proper output addresses due to repetition of the entries in the row Although this scheme achieves considerable savings compared to LMS, there is still duplication of information in the *BGT* entries because even now, each table is required to store information for each connection passing through the switch, i e. one entry per connection in each *BGT* Thus, if V is the maximum number of connections allowed to pass through a port, each *BGT* contains one entry each for all the NV connections

Table for BGT in LMS

Virtual Connection	Destination Channel Group						
	1	2	3	4		N-1	N
0	a	b	c				
1	d						
2	e	f	g	h			
NV-1	x	y					

Modified BGT (GVT) as in [33]

Virtual Connection	Destination Channel Group						
	1	2	3	4		N-1	N
0	a	b	c	a		a	b
1	d	d	d	d		d	d
2	e	f	g	h		e	f
NV-1	x	y	x	y		x	y

Figure 2 3 Table Organization in LMS and in modification proposed by Min et al in [33]

passing through the switch Apart from that, the table management for dynamic group changes becomes a problem, because even for a single addition or deletion from the group, the whole GVT changes and hence, all the *BGTs* have to be modified

2.3 Reducing the Memory Requirement of LMS

In this section we present a modification to the LMS to eliminate the table duplication problem and achieve the minimum theoretical limit of the memory requirement The switch is referred to as *Memory Optimized Lee Multicast Switch (MO-LMS)*, in further discussions With this architecture, apart from saving on memory requirement, the use of an expanded copy network is feasible to improve throughput and overflow probability performance

In MO-LMS, we solve the memory duplication problem by avoiding the need for the table at the output of the copy network At the output of the copy network the table is required primarily to supply routing information to the copies of the incoming cell (the output port number) Control information like priorities and outgoing VCI/VPI numbers can be added at the output of the routing network Thus, in MO-LMS, we split the *BGT* into two parts, the Routing Table (*RT*) at the input of the switch, containing the routing information and the Connection Control Table (*CCT*) at the output, supplying the control information

At the input of the switch a routing header is attached to the incoming cell after

which it is fed to the copy network The copy network in MO-LMS is a modified version of Lee's Copy Network to have the copy contain the routing information at the output This routing information is generated as the cell passes through the copy network. The control information for the individual copies is stored in the CCT at the target output port Although the copies of the cells may appear at any output of the copy network no information need be stored at the outputs because the routing information is now available in the routing header of the copy

We now describe the details of the modifications At the input of the switch, routing information is obtained from RT The cells are appended with an N bit routing header (for an N-output-port switch) Bit i of the routing header is 1 if a copy needs to be routed to output port i of the switch The address interval for the copy network is added, as in the LMS, by the Running Adder and Dummy Address Encoders The active cells are compacted and fed to the Replicating Banyan Network (RBN) which processes the $N-$bit routing header in addition to performing the functions of the Broadcast Banyan Network as has been described by Lee [27]

Supppose that a node in stage k recieves a packet from a node in stage k-1 with the copy network header containing an address interval specified by the two binary numbers min (k-1) $= m_1 m_2 \quad m_n$ and max (k-1) $= M_1 M_2 \quad M_n$, where $n = log_2 N$

The algorithm followed by each individual RBN element is as follows (See Figure 2 4)

1 If $m_k = M_k$, then send the cell out on link 0 or 1 respectively The routing header is not modified

2 If $m_k = 0$ and $M_k = 1$, then the cell is replicated and the original cell is sent on both the links with the headers modified as below

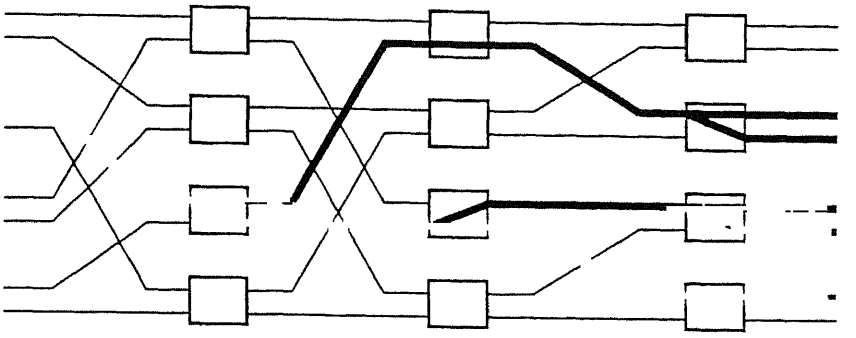
The header determining the address interval is modified by splitting the original address interval into two subintervals, according to the boolean interval splitting algorithm as determined by the following expression

min (k) = min (k-1) $= m_1 m_2 \quad m_n$

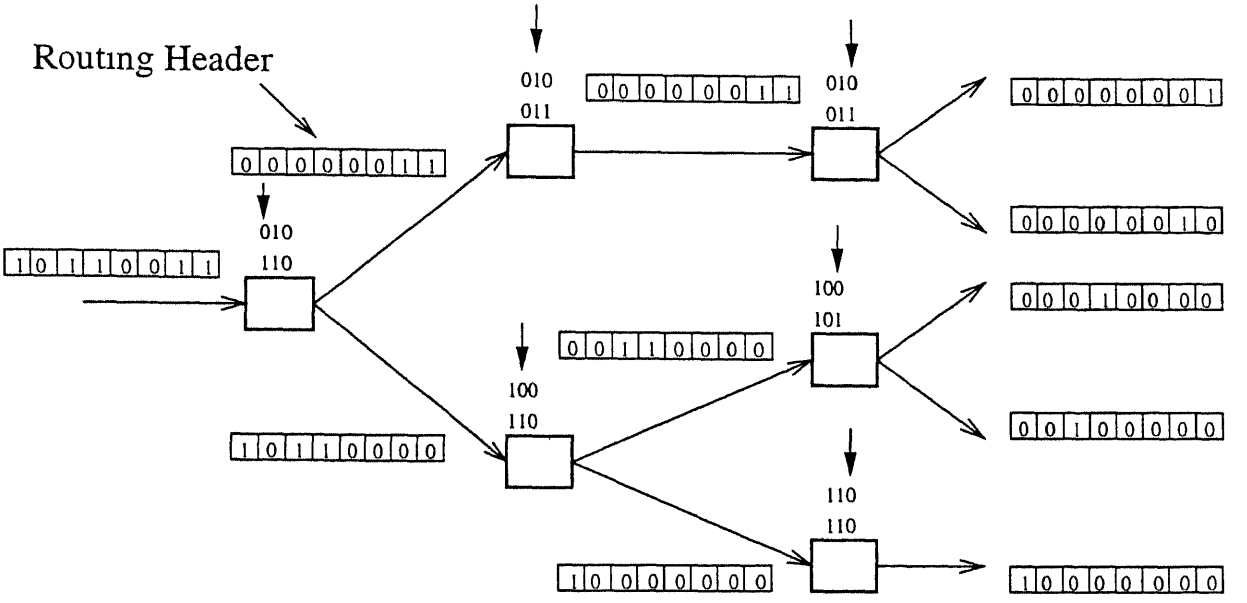
max (k) $= M_1 M_2 \quad M_{k-1} 01 \quad 1$

for the packet sent out on link 0, and

min (k) $= m_1 m_2 \quad m_{k-1} 10 \quad 0$



(A)



(B)

Figure 2 4 Header processing by the RBN elements

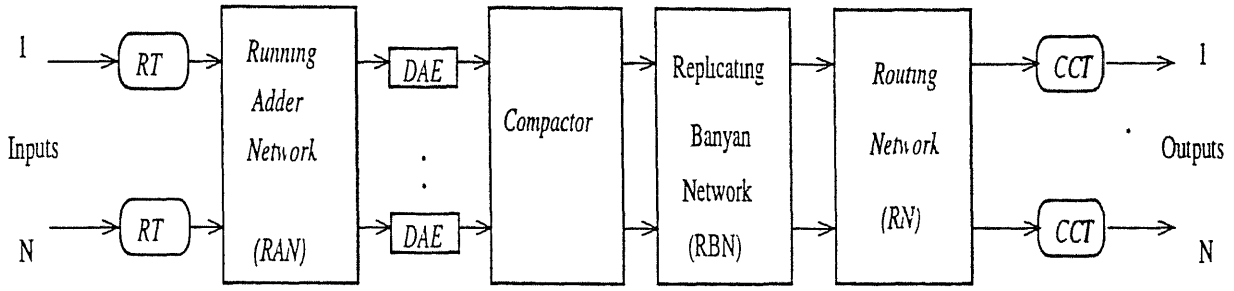


Figure 2 5 Block Diagram of the Memory Optimized Lee Multicast Switch (MO-LMS)

max (k) = max (k-1) = $M_1 M_2 \quad M_n$

for the packet sent on link 1 at stage k

Routing headers of the two copies are derived from the original routing header as follows

2A Bits that are 0 in the input routing header remain 0 in both the output routing headers

2B. Bits that are 1 in the input routing header are distributed to the output routing header according to the address interval splitting done at the element, i e , if the input cell has p bits set to 1 in its routing header and the cell going to the upper output has q copies embedded in it (the number of copies embedded in any intermediate cell is determined by the address interval assigned to it, following Lee's boolean interval splitting algorithm), the routing header attached to that particular copy of the cell will have q bits set to 1 and the remaining bits will be 0. These q bits can be selected from among the p bits which were 1 in the routing header of the input cell arbitrarily The routing header of the other copy of the cell will be 0 in all the bits except at the remaining $p - q$ bits, which were set in the input routing header

At the output of the copy network, the individual copies have a routing header with just one bit set to 1 The position of this 1 determines the destination output port for the copy Thus no table is required at the output of the copy network The VCI/VPI translation is done at the output of the routing network using *CCT* As is shown in the next section, the memory requirements of an $N \times N$ MO-LMS supporting up to V

simultaneous VCs per port is $V(N + W)$ bits per port which is actually the theoretical minimum required to completely describe all the combinations. The architecture of the MO-LMS is illustrated in Figure 2 5

2.3.1 Memory Savings

Gupta has shown that the hardware complexity of the multicast ATM switch will primarily be determined by the memory elements [14] A major contribution to this memory comes from the connection control tables Therefore, we are using it as a basis of comparison for the hardware complexity of the two architectures

Assuming an upper limit of V on the number of simultaneous VCs that can be supported per port of the switch and a control information of W bits per output VC, the memory requirement for the two schemes can be calculated as below

Lee Multicast Switch

In LMS (Figure 2 1), there is a table (CT) at each input, which determines the number of copies required for a cell belonging to a particular input VC. This requires $\log_2 N$ bits LMS also contains tables ($BGTs$) at each of the outputs of copy network, each containing control information of W bits along with $\log_2 N$ bit routing information, for all the copies of a VC This corresponds to a storage requirement of $\log_2 N + W$ bits per entry Each BGT contains information about all the copies belonging to all the connections passing through the switch As each output port can handle at most V VCs simultaneously, the sum total of the number of entries allowable can not exceed NV Thus the number of entries in each BGT is NV Thus the total storage requirement for the LMS becomes

$$V[\log_2 N + N(W + \log_2 N)] \text{ bits/port}$$

Memory Optimized Lee Multicast Switch

In MO-LMS (Figure 2 5) there is a routing table (RT) at each input, containing the N bit routing information about each input VC The number of copy information can also be derived from it The CCT at each output of the switch, contains W bits of information about each VC belonging to that output port The total storage requirement for the MO-LMS becomes

$$V[N + W] \text{ bits/port}$$

It should be noted that for these calculations, we have not put any restrictions on the

maximum fanout of a connection Any connection can be admitted provided the limits on the maximum simultaneous VCs is not violated at any of the ports

As is clear from the expressions of memory requirements, the MO-LMS requires a memory that is orders of magnitude less than that of the LMS For example, a 256×256 switch will require 36 9 Mbits/port for the LMS while for MO-LMS only 888 Kbits/port will be required Here, we have assumed that the switch supports a maximum of 3000 simultaneous VCs per port and that there are 40 control bits per connection Note that our scheme achieves the theoretical minimum of the memory requirement

Another way to compare the memory requirements of the two schemes is to compare the memory occupied by a multicast connection, with a fanout of F The total memory occupied by the connection in the LMS is $[\log_2 N + NF(W + \log_2 N)]$ Even after modifications suggested in [33], the memory occupied by the particular connection is $[log_2 N + N(W + log_2 N)]$ The same connection will require just $[N + FW]$ bits in the MO-LMS Considering the example of the same 256×256 switch, a multicast connection with a fanout of 128 will require 1 57 Mb and 12 296 kb in LMS and modified LMS incorporating the modifications suggested in [33] respectively as compared to just 5376 bits in MO-LMS, assuming a control information of 40 bits The advantage of MO-LMS over modified LMS increases for lower fanout values Note that although the modifications suggested in [33] lowers the memory occupied by a particular connection, the total memory required for the switch, being determined by the maximum number of allowable connections passing through the ports, still remains the same

2.4 Reducing Head Of Line Fanout Blocking and Overflow in Space-division Multicast Switches

A limitation of the LMS is that a single connection with a large fanout can prevent most cells including itself that enter during a given cell time from passing through the copy network [31, 44] For example, in a situation in which the cell at input 1 has a fanout of 1 and the cell at input 2 has a fanout of N, the net throughput of the switch during that cycle will be only 1 cell This is called *Head Of the Line Fanout (HOL-FO) Blocking* in [31] One way to overcome HOL-FO blocking is to use an asymmetric copy network

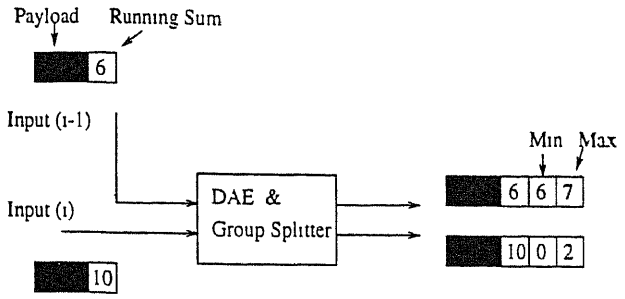


Figure 2 6 Group-Splitting in an 8 X 8 switch

with the number of output ports greater than the number of input ports For example, Turner shows that using an $N \times 2N$ copy network one can get a sustained throughput of N cells/cycle [44] It should be noted that such a solution was not practical earlier as the table duplication increased k-fold for an $N \times kN$, $k > 1$ 0. LMS However with the MO-LMS, such an asymmetric copy network can be easily used to reduce HOL-FO blocking A second way, the one that we follow in our architecture, is known as *group-splitting* In this approach whenever a cell can not be fully served in a particular time slot, the group of destinations is split into two parts The copies that can be made in the time slot, are made using the original cell and the remaining copies are made in the next time slot, using a copy of the original cell stored for the purpose Group Splitting is shown in Figure 2 6

A second limitation with the LMS is the *overflow* In a given slot, when the total number of copies to be produced exceeds the size of the copy network, overflow occurs and the excess cells are dropped In the LMS, the overflow is detected by the dummy address encoders, and the overflow cells are dropped Overflow puts severe limitations on the throughput that can be achieved by the copy network To reduce the overflow probability, various means have been suggested in the literature

The suggestion of splitting the unicast and multicast traffic by Gupta [14] can be used to reduce the overflow as the separation of the unicast traffic reduces the load on the copy network

To reduce the overflow probability, Lee suggests the use of an expanded copy network (Figure 2 7), followed by an expanded routing network [27] This suggestion is not usable with LMS because increasing the number of output ports, increases the number of tables required and hence further exacerbates the table duplication problem However,

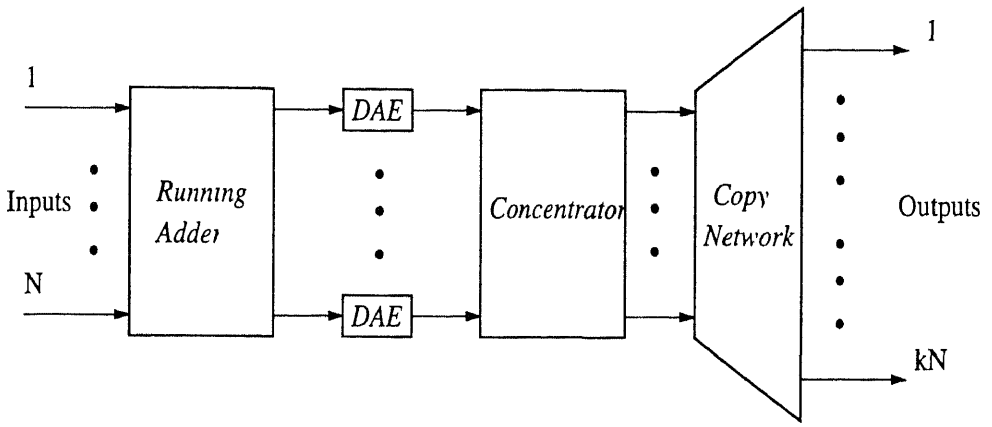


Figure 2 7 Overflow Reduction using Expanded BBN

since the MO-LMS eliminates the table duplication problem, this suggestion can now be implemented in practice The chief disadvantage of using the expanded copy network is due to increased cost of the VLSI realization Most of the space division part of the switches are pin constrained in terms of the VLSI implementation Thus if we use the expanded copy network followed by an expanded routing network, the chip count for this logic is approximately doubled

Instead of expanding the output of the copy network, we can expand the input of the copy network (Figure 2 8) The excess inputs are used as recirculation ports The dummy address encoders are modified such that they filter out the overflow cells and feed it to a recirculation concentrator The concentrator concentrates the overflow cells and stores them in the buffers These buffers contain the cells for the recirculation ports for the next slot The number of buffers required is equal to the number of recirculation ports The sequencing is maintained by giving higher priority to the recirculated cells In fact, as the running adder provides inherent priority arrangement, all that is needed is to feed back the recirculated cells at the upper ports of the copy network Group splitting can also be used to improve the throughput The VLSI requirement of this scheme is lower than that of the expanded copy network scheme, because it does not require an expanded routing network This point will be elaborated upon in the next subsection

Liew [29] suggests use of speeded up copy network or a number of copy networks operating in parallel to reduce the overflow

Instead of dropping the cells, the cells can be buffered at the input for retry in the next

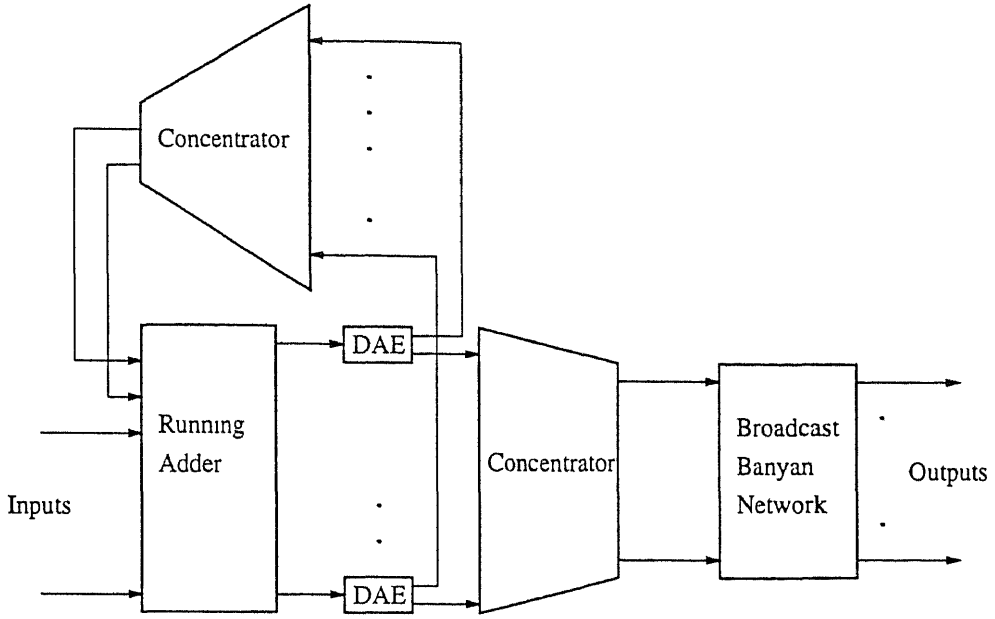


Figure 2 8 Copy Network with Recirculation for overflow reduction

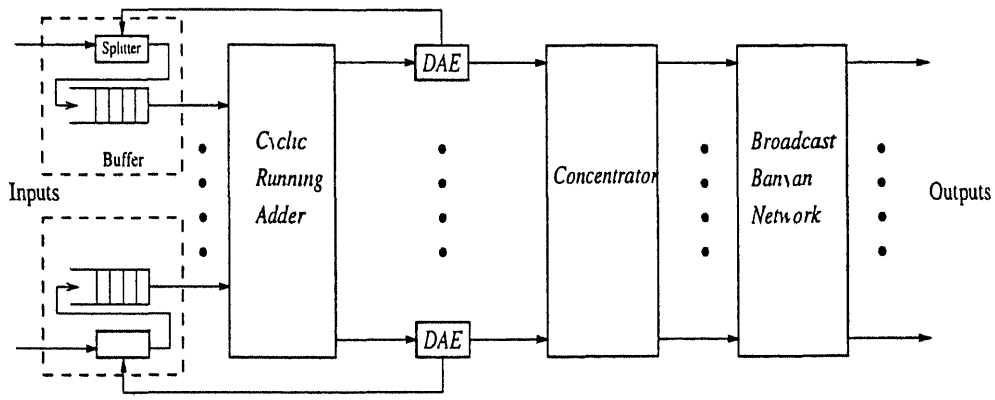


Figure 2 9 Input Buffered Copy Network

exceeds the number of output ports of the copy network, the cells for all the input ports including and following the port whose cell caused the overflow need to be transmitted in the next slot This approach, combined with the group splitting is suggested in [28] (Figure 2 9) It uses buffers at the input and a feedback is provided from the output to the input to determine whether splitting is required The running adder always starts the running sum calculation from the input port subjected to group split in the previous slot

Input buffering, apart from inefficient buffer utilization, also suffers from HOL blocking, limiting its throughput A better way is to share the buffers at the output Note

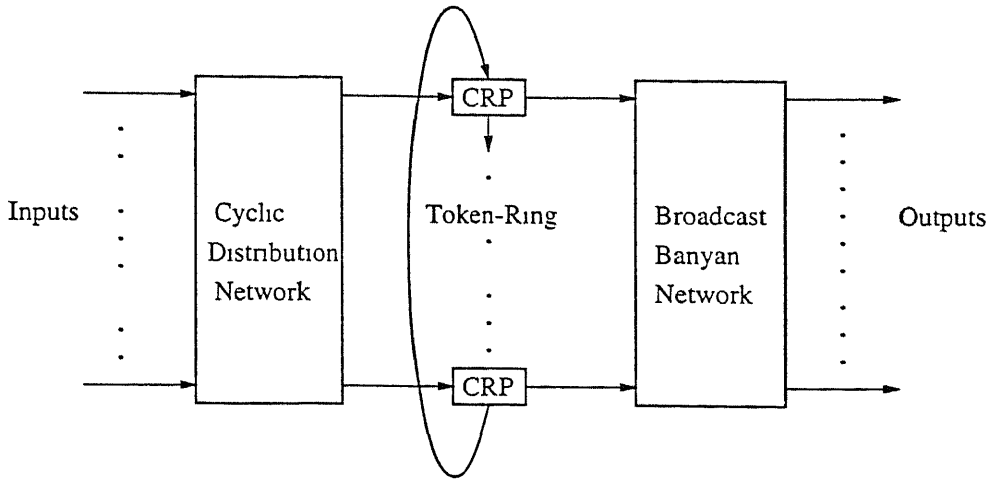


Figure 2 10 Output Buffered Copy network suggested in [49]

that the copy network with recirculation provides an indirect shared output buffering in the form of recirculation buffers

Zhong et al [49] suggest a shared output buffering scheme for reducing the overflow probability (See Figure 2 10) It also uses the concept of group splitting to achieve higher throughputs It uses the nonblocking property of the broadcast banyan network, subjected to a cyclically compact, monotonic address intervals The shared buffers are distributed and are managed by a cyclic distribution network and contention resolution processors (CRPs) It uses a token-ring reservation scheme to schedule the cells in the buffer, such that only the cells which can be served in a particular slot are passed to the BBN It also performs the group splitting in case of overflow and schedules the two cells in two consecutive cell times This architecture has scalability problems due to the sequential token-ring reservation algorithm used

Liu and Mouftah [31] also suggest a scheme for shared buffering of the overflow cells It uses on-line departure scheduling and distributed shared memory buffering It also relies on dynamic group splitting to achieve higher throughputs To guarantee a first in first out service, the departure time of the inputs are scheduled upon their arrivals An information called Waiting Time (WT) is added to the cell, which indicates the number of time slots the cell is supposed to wait in the buffer before proceeding to the BBN In the buffer, the WT of each cell is decremented every slot and when it becomes zero, the cell is passed on to the BBN Since the buffer manager has to decrement WT of all the cells in the buffer it can become a bottleneck in the realization of the switch In

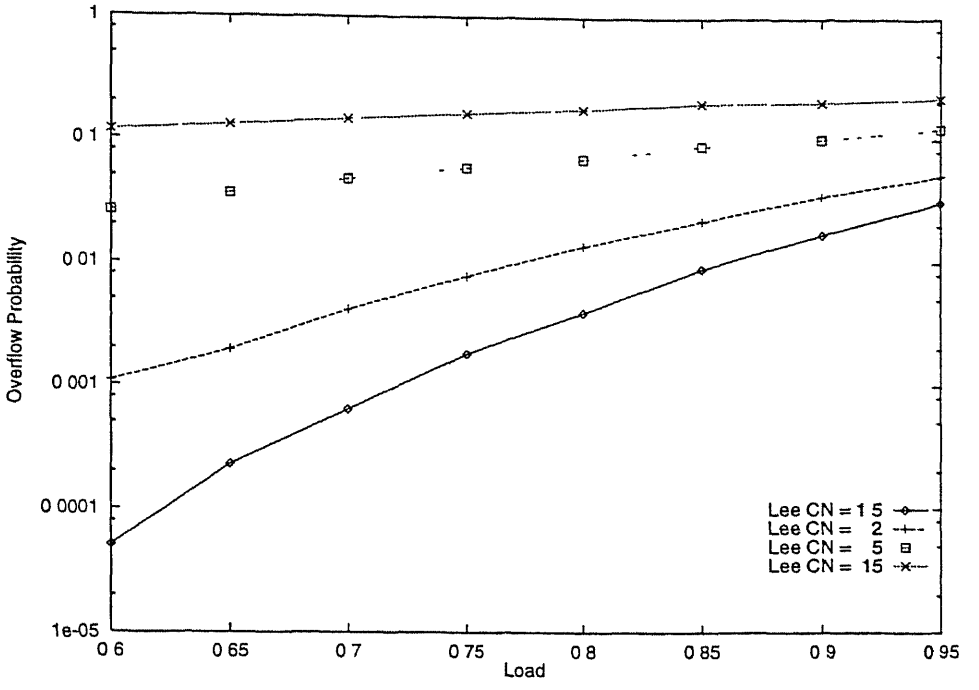


Figure 2.11 Overflow Performance of Lee's Copy Network

section 2.5, we propose our architecture with a mechanism to avoid this bottleneck.

2.4.1 Comparison of overflow reduction schemes

In this section, we will compare the performance of different overflow reduction schemes using simulation models. The input traffic process for the simulation is Bernoulli with mean p. The number of copies for an arriving cell is in the range 1 N and is obtained from the truncated geometric distribution

$$P_k = \frac{(1-q)q^{k-1}}{1-q^k} \qquad 1 \leq k \leq N$$

where, P_k is the probability that an input cell wants k copies and q is a parameter which is a function of the average number of copies (CN) required by the incoming cells and is given by

$$CN = \frac{1}{1-q} - \frac{Nq^N}{1-q^N}$$

The overflow characteristics of the Lee's copy network is shown in the Figure 2.11 and that of the improved schemes are shown in Figure 2.12, Figure 2.13, Figure 2.14 and Figure 2.15 respectively. The switch size is assumed to be 64×64 for all the simulations

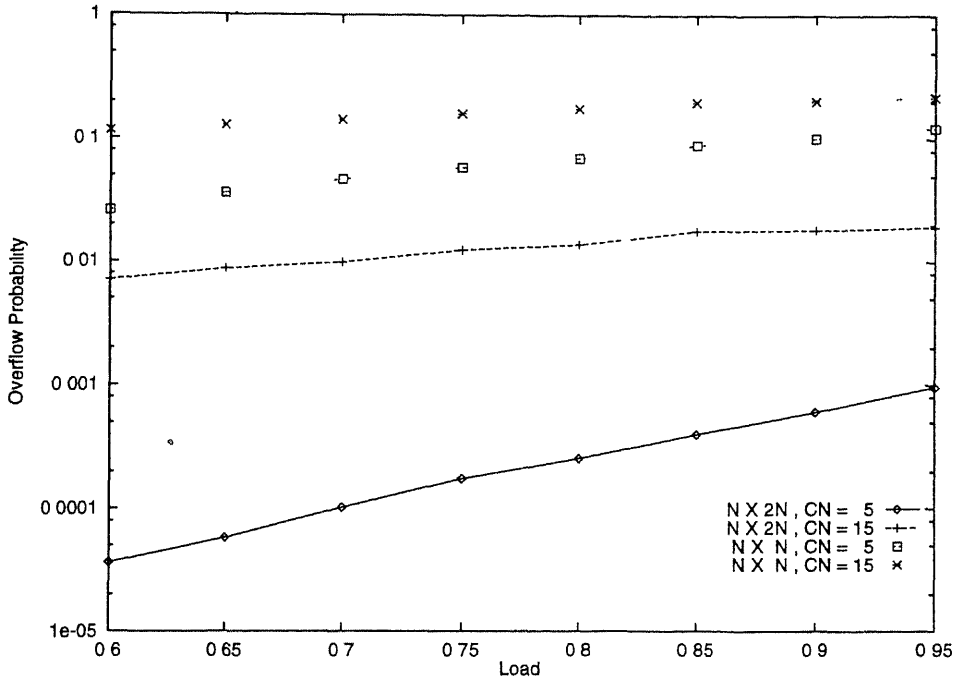


Figure 2 12 Overflow Performance of Expanded Copy Network

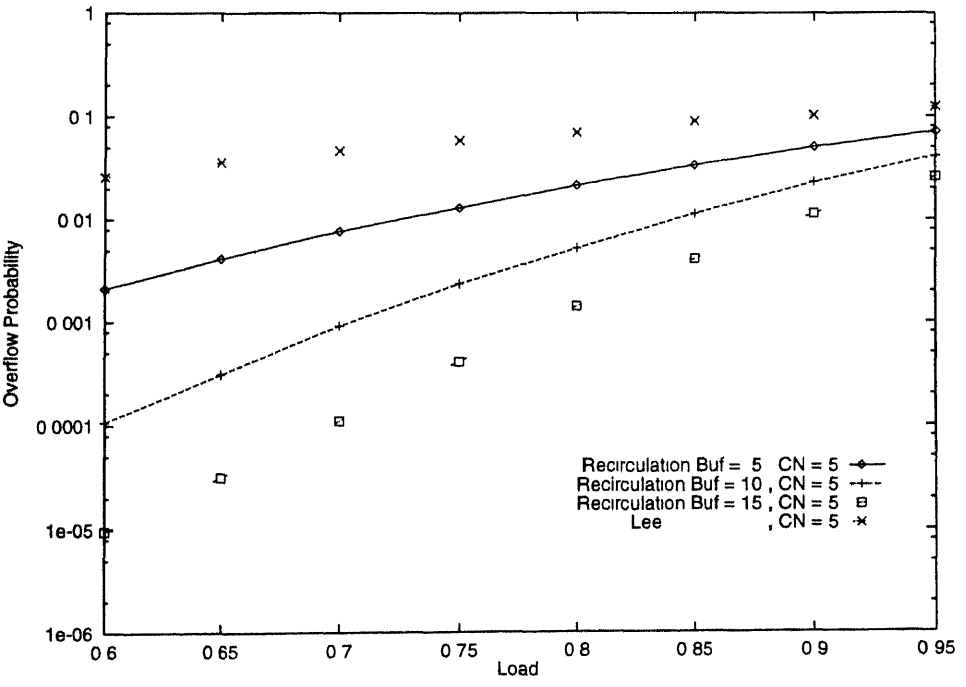


Figure 2 13 Overflow Performance of Recirculation Copy Network

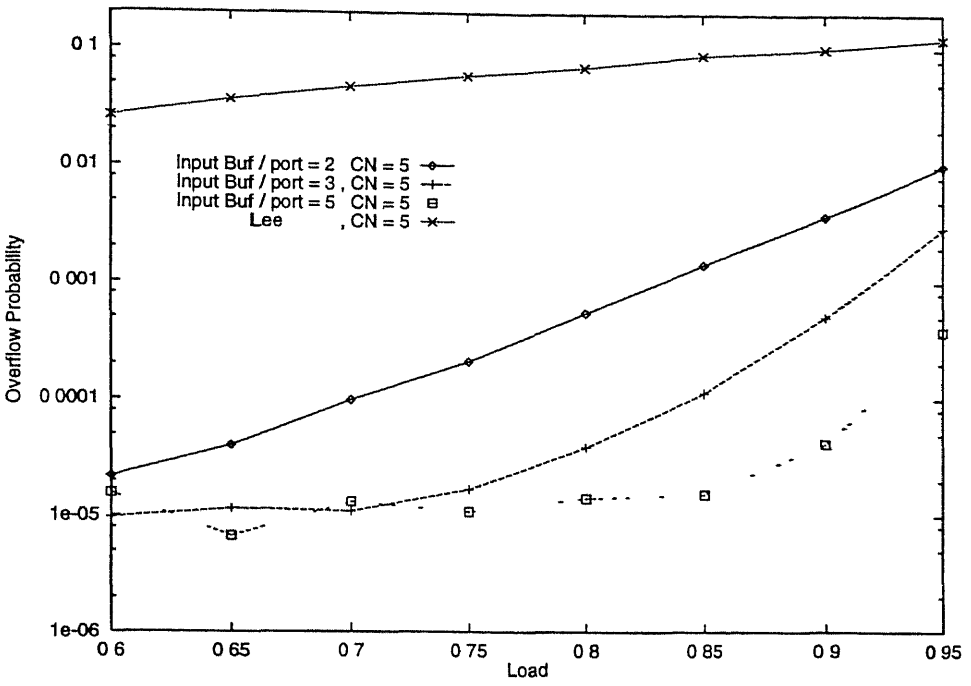


Figure 2 14 Overflow Performance of Input Bufferd Copy Network

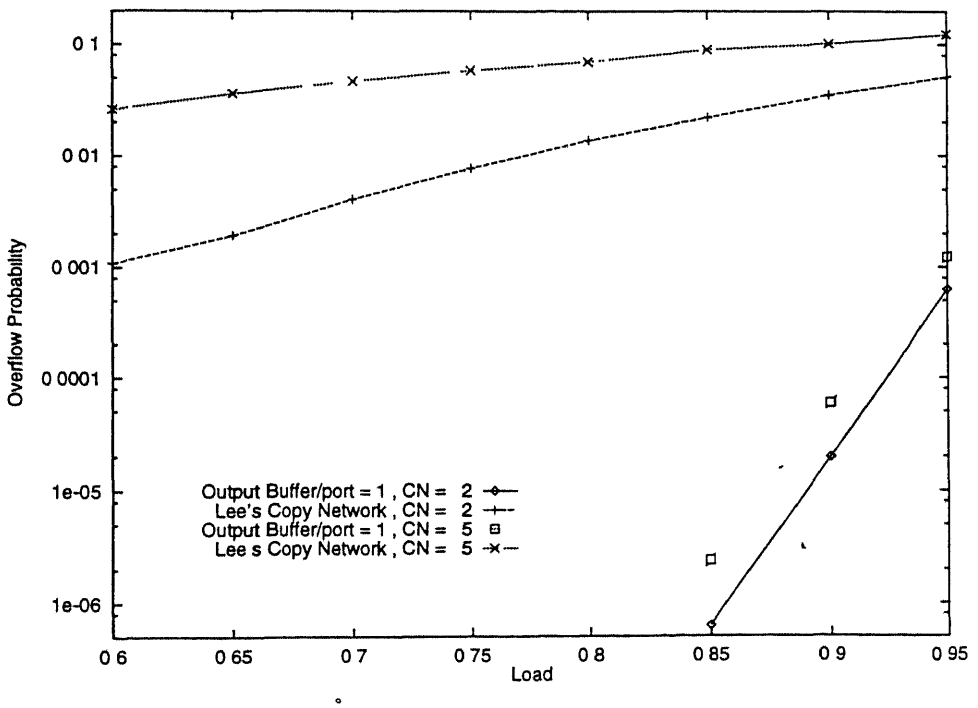


Figure 2.15 Overflow Performance of Output Bufferd Copy Network

As can be seen from the graphs, the expanded copy network and the input buffered copy network have very high overflow probabilities For example, for 80 % load with average number of copies being 5, the overflow probability for the $N \times 2N$ copy network is 4 03 X 10^{-3} and that for the input buffered copy network with a single buffer per input port is 1 44 X 10^{-3} The output buffered and recirculating copy networks provide adequate performance For example, the overflow probabilty of both the schemes assuming the same traffic, even for 1 buffer/port is small enough to be measured in our simulation However, the recirculating copy network requires a complicated running adder, DAE and concentrator due to the increased size for accomodating the feedback inputs In view of the pin limitations of the VLSI fabrication, this would mean a significant increase in the chip count required for realization of the recirculating copy network Therefore, the shared output buffered copy network is preferred over the others We use this approach in our proposed architecture

2.5 A Space Division Multicast Switch Architecture

In this section we describe our copy network architecture, based on the extended nonblocking multicast condition [31] The salient features of our architecture are mentioned below

Nonblocking

Like Lee's Copy network, the proposed copy network is also nonblocking This means that as long as there are at least N copies to be produced in a slot, the throughput of the copy network will be N copies per cell time

Minimum memory reqirements for the connection control information

MO-LMS architecture is used to minimize the memory requirement required for maintaining the connection control information The memory required in this architecture is equal to the theoretical minimum

Dynamic Group Splitting

The proposed architecture uses dynamic group splitting to improve the overflow performance If all the copies of a particular input cell cannot be made in one cell

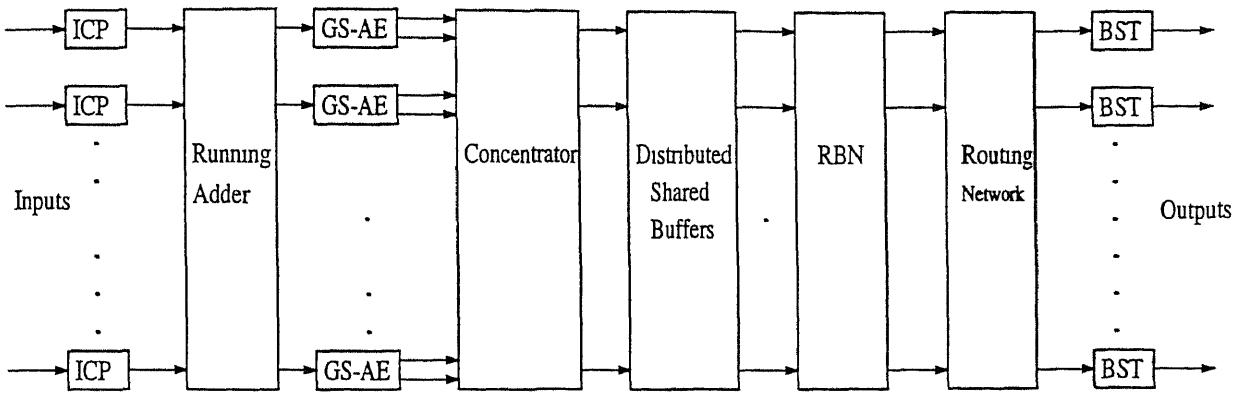


Figure 2 16 Proposed Space-Division Multicast Switch Architecture

time, only the allowable copies are made and the remaining copies are made in the next cell time Group splitting, apart from reducing the overflow, also increases the sustained throughput of the copy network, as the HOL-FO blocking does not occur Moreover, it also ensures fairness among the input ports

Distributed Shared-Memory Buffering

To reduce the overflow, the proposed architecture uses the shared buffering mechanism as in [31] The memory consists of a number of buffers operating in a distributed manner to achieve effective shared buffering without the need of a single buffer shared among all the ports The use of a shared buffer for overflow reduction leads to a better buffer utilization

On-line departure time scheduling

In order to guarantee first in first out delivery, the proposed architecture uses an on-line departure time scheduling as in [31] The cells are tagged with a number called Exit Stamp (ES) on arrival to the switch This number is later used by the buffer manager to schedule the exit of the cells from the shared buffers

The block diagram of the proposed space-division multicast switch architecture is shown in Figure 2 16 The design details of the individual blocks are explained below

Input Cell Processor (ICP) The input cell is first fed to this block, which finds out the routing information corresponding to the incoming cell from the routing table (RT) and appends this information to the cell as the routing header The

information in the RT is indexed by the incoming VCI/VPI and consists of a bit pattern, denoted as Routing Header consisting of $\log_2 N$ bits for an N output port copy network. This routing header contains information about the destinations to which the copies of the cell are to be routed. The information regarding the number of copies required to be made, can either be calculated from this routing header or to save the computation overhead, can be computed by the connection control process and put in the RT along with the routing header. Both the routing header and the information regarding the number of copies are appended to the cell as a *copy network header*.

Running Adder Network The number of copies as specified in the copy network header is used by the Running Adder network to generate running sums of copy numbers. The maximum copies required by any cell is N. Hence the maximum running sum is less than or equal to N^2. Therefore at most $2 \log_2 N$ bits are required for the sum calculated by the running adder. Thus the running adder calculates a number that is modulo N^2 running sum of the number of copies with an offset equal to the feedback number from the previous slot. This number is divided into two parts - the most significant $\log_2 N$ bits constitute the Exit Stamp (ES), while the least significant $\log_2 N$ bits is called the Running Sum (RS). The running adder replaces the field indicating the number of copies information with these two fields as shown in Figure 2.17.

Group-Splitting Units and Address Encoders (GS-AE) Using ES and RS of consecutive ports, the following algorithm decides on the need for group splitting at the port and also assigns the dummy address intervals for the replicating banyan network (RBN).

- if $ES_i = ES_{i-1}$, no group splitting, the input cell is forwarded to register 1 (R1) with the following fields appended to it
  - ES $= ES_i$
  - min $= RS_{i-1}$
  - max $= RS_i - 1$
  - A $= 1$ if $RS_i > RS_{i-1}$ else A $= 0$

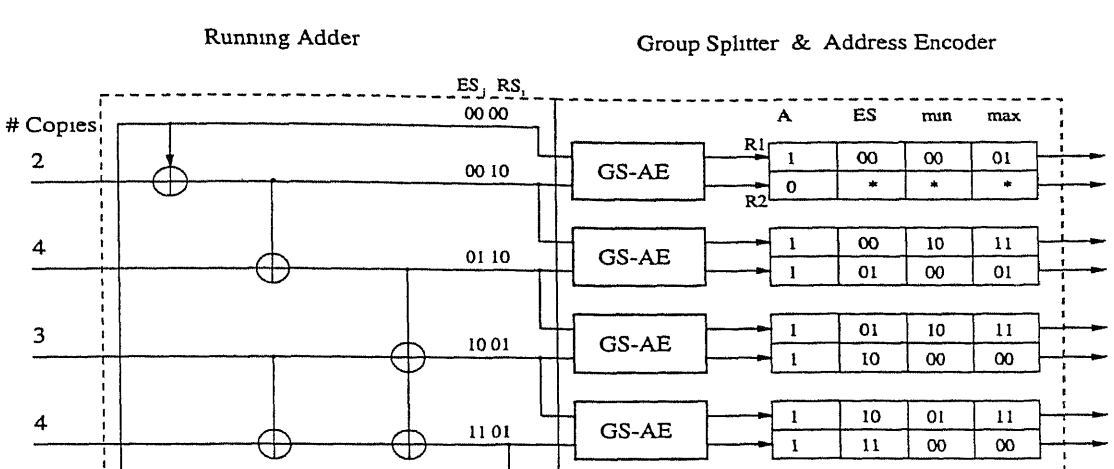


Figure 2 17 Running Adder, Group-Splitter and Address Encoders

The A field of the cell in the register 2 (R2) is set to 0

- if $ES_i > ES_{i-1}$, group is split and the two split cells are forwarded to registers 1 and 2 respectively

For the cell forwarded to R1

ES $= ES_{i-1}$

min $= RS_{i-1}$

max $= N - 1$

A $= 1$

For the cell forwarded to R2

ES $= ES_i$

min $= 0$

max $= RS_i - 1$

A $= 1$

Here,

min denotes the minimum address in the dummy output address set

max denotes the maximum address in the dummy output address set

ES refers to Exit Stamp determining the waiting time for the cell

A indicates whether the input is active or idle

A *FeedBack Unit (FBU)* is responsible for generating the Feedback signal (FB) fed to the uppermost input of the running adder The feedback signal consists of two parts, ES_f and RS_f, which are determined as follows

$ES_f = ES_N$ and $RS_f = RS_N$ if $BE = 0$ or $BE = 1$ and $ES_N > 0$

$ES_f = 0$ and $RS_f = 0$ if $ES_N = 0$ and $BE = 1$

Here, BE is a flag, that is used to indicate the buffer status at the end of previous slot $BE = 1$ indicates an empty buffer The feedback signal is delayed till the next time slot

Concentrator and Shared-Memory Buffering Stage

This stage consists of a cascaded combination of a $2N \times 2N$ running adder, a $2N \times 2N$ reverse banyan network and N distributed buffers (See Figure 2 18) It performs two functions, cell concentration and buffering The running adder adds up all the active cells (specified by the A field of the cell) to form $2N$ queueing addresses The bottom running sum is fed back to the top line of the running adder at the end of each time slot as the starting point of the next round of calculation Therefore, the queueing addresses calculated by the running adder are compact and cyclically sequential Physically the shared memory buffer consists of N individual buffers at the output ports of the reverse banyan network. The reverse banyan network routes the cells cyclically to the queues. In order to multiplex the $2N$ inputs to the N interleaved queues cyclically, the switching elememts in the last stage are modified to be 2 to 1 time division multiplexers, each one of which delivers the two input cells to the output port one by one In order to maintain the sequence, the cell with the smaller input port address is given higher priority, as it may happen that the consecutive inputs from the same input port be delivered in the same time slot A Sequence Generator is used to schedule the exit of the cells from the buffer to the RBN It generates a $\log_2 N$ bit number during each cell time Only the buffers whose head of the line cell has the ES field matching with the index generated by the index generator, are allowed to forward their head of line cell to the RBN This ensures that the cells that appear at the input of the RBN are cyclically compact and the output addresses are cyclically monotone, with the total demand of copies not exceeding N The sequence generator acts as a modulo N counter which is

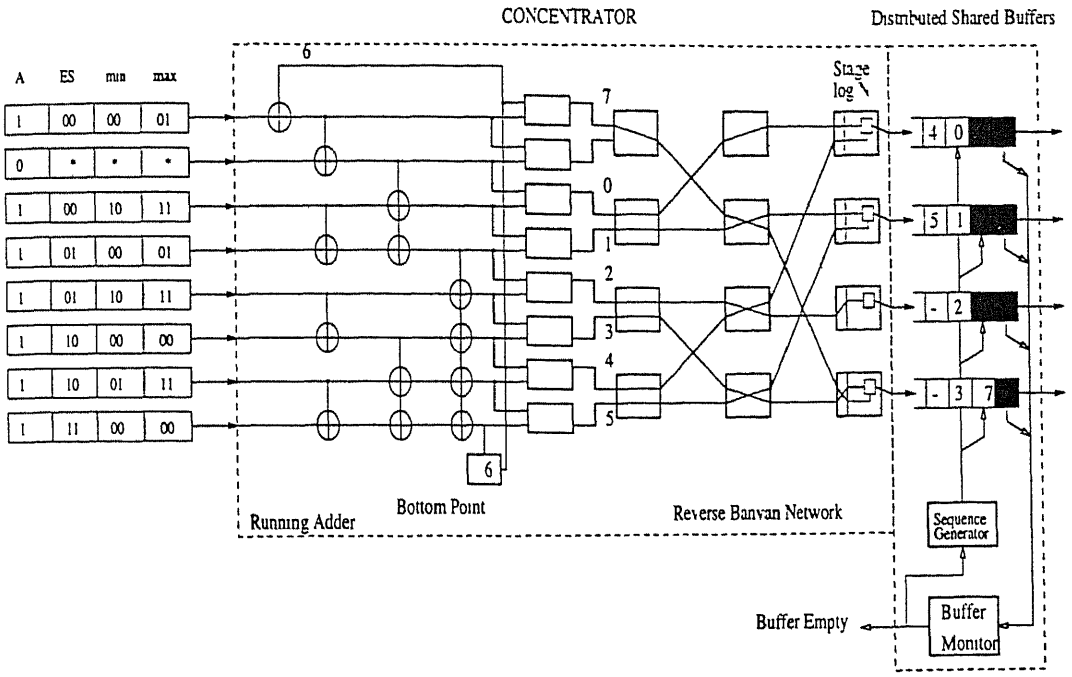


Figure 2 18 Concentrator and Distributed Shared Buffering Stage

reset to zero if the buffer was empty at the end of the previous slot

Replicating Banyan Network (RBN) It is a simple banyan network with each Switching Element capable of either routing an input cell to one of the two output ports or replicating the incoming cell to both of its outputs based on the two bit information in the header It replicates cells according to a Boolean Interval Splitting algorithm based on the dummy address interval in the copy network header The RBN elements also process the routing header attached to the cell according to the algorithm given in the section 2 3 As the inputs to the RBN are compact with cyclical shift and the outputs are monotone with cyclical shift, it is nonblocking

Routing Network (RN) The routing network routes the cells appearing at the output of the RBN to appropriate output port of the switch An output buffered switch is required to achieve maximum possible throughput These cells are stored in the output buffer

Output Buffer-Scheduler-Translator (BST) The cells from the RN are collected in the output buffer. The dimensioning of the output buffer depends on the scheduling algorithm employed The scheduler schedules the exit of the cells from the buffer

depending on the Quality of Service (QoS) contract established at the beginning of the connection To aid this function, the connection control table (CCT) has a resource provisioning field in it The scheduled cell is transmitted to the output after performing the VCI/VPI translation in the header The CCT has entries indexed by the incoming VCI/VPI, containing the information regarding the outgoing VCI/VPI and the resource provisioning field, flow control information, priority etc for scheduling the cell, as required by the scheduler

2.6 Conclusion

As we have seen in this chapter, the proposed architecture brings about considerable reductions in the memory requirements for supporting multicast in ATM switches, as compared to existing architectures The modification incurs marginal increase in complexity due to additional processing of the routing header by the RBN elements, which is more than offset by the large memory savings achieved A small speedup of the fabric is required to carry the cell within the same time duration, as it also contains the routing header in addition to the header fields present in the LMS For example, a 64×64 switch will need to be operated at a speed that is 13 12% higher than the LMS For very large switches, this speedup can become prohibitive However, in chapter 5, we suggest a method to construct large multicast ATM switches using the MO-LMS, without excessive speedup

Since the memory size dominates the multicast switch hardware complexity, the MO-LMS makes practical implementation of multicast switches for ATM possible at a hardware complexity that is substantially less than that in the existing architectures

Apart from optimizing on the memory requirement for the connection control information, our architecture improves on the throughput performance using shared memory buffering and group-splitting The overflow performance of the proposed architecture are the same as that of the output buffered copy network shown in Figure 2 15 The figure shows that even a single buffer per port reduces the overflow probability to very small values

Chapter 3

Channel Grouping

3.1 Introduction

In a single channel ATM switch a cell is switched from a particular input port to its destination output port In a multichannel switch output ports are grouped and the cell is switched to any output port belonging to an appropriate channel group Throughput in single channel switches can be severely limited by internal blocking, output contention and head of the line Blocking A variety of buffering strategies have been proposed to increase the throughput Multichannel switches exploit the concept of channel grouping to achieve higher performance The advantages of statistically sharing a high channel capacity under such conditions are well known For a given input traffic intensity, a larger channel group size is less likely to experience blocking for a single ATM cell, for a burst of cells, or for a connection request Other performance measures such as delay, buffer overflow, and congestion would improve when multiple channels are grouped together as a single resource

3.2 Group Knockout Principle

A fundamental concern in designing an $N \times N$ switch is that up to N input cells may arrive simultaneously destined for the same output Given that each output port can deliver only one cell at a time, the multiple arrivals must be buffered If arrivals on different input lines are statistically independent, then the probability of a more than L

arrivals decreases rapidly with L, even for arbitrarily large N Therefore many switch designs exploit this property by allowing upto L cells for a given output port in a slot, and drop the excessive cells [46, 6, 13] The value of L is chosen such that the dropping probability will be sufficiently below all other loss mechanisms (buffer overflow, link failure etc) This design principle was first used in the knockout packet switch and is hence called as the *Knockout Principle* [46]

The following analysis is given in [8] If the cells have independent, uniform destination address distributions, then the probability that k cells arrive for a particular output is

$$P_k = \binom{N}{k} \left(\frac{p}{N}\right)^k \left(1 - \frac{p}{N}\right)^{N-k} \tag{3 1}$$

where p is the probability that a cell arrives on an input line in a slot (i e switch loading) The probability for a cell to be dropped is

$$Pr(PacketLoss) = \frac{1}{p} \sum_{k=L+1}^{N} (k - L) \binom{N}{k} \left(\frac{p}{N}\right)^k \left(1 - \frac{p}{N}\right)^{N-k} \tag{3 2}$$

As $N \to \infty$, (3 2) becomes

$$Pr(PacketLoss) = \left[1 - \frac{L}{p}\right] \left[1 - \sum_{k=0}^{L} \frac{p^k e^{-p}}{k!}\right] + \frac{p^L e^{-p}}{L!} \tag{3.3}$$

The important point is that $L = 8$ is large enough to keep the cell loss probability below 10^{-6} for any switch size with $p = 0\ 9$ (i e 90 % load) under uniform independent traffic assumption Every increment in L further reduces the cell loss probability by approximately an order of magnitude Further, the value of L required to achieve a given loss probabilty is not very sensitive to switch loading

The knockout principle can be generalized to a group of outputs Instead of just a single output, we consider a group of G outputs and treat them as if they were one output Then using the same assumption as above, the probability that an input cell is destined to this group of outputs is simply G/N If we allow only m cells to pass through to the group outputs, then (3 2) becomes

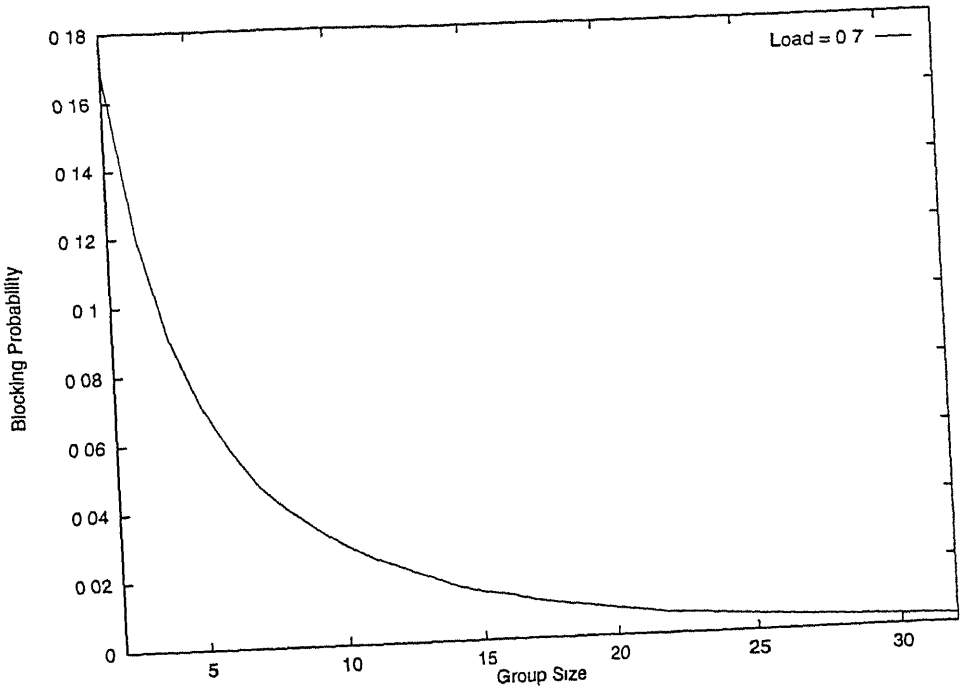


Figure 3 1 Effect of Channel Grouping for m = G

$$Pr(PacketLoss) = \frac{1}{Gp} \sum_{k=m+1}^{N} (k-m) \binom{N}{k} \left(\frac{Gp}{N}\right)^{k} \left(1 - \frac{Gp}{N}\right)^{N-k} \quad (3.4)$$

As $N \to \infty$,

$$Pr(PacketLoss) = \left[1 - \frac{m}{Gp}\right] \left[1 - \sum_{k=0}^{m} \frac{(Gp)^k e^{-Gp}}{k!}\right] + \frac{(Gp)^m e^{-Gp}}{m!} \quad (3.5)$$

Figure 3 1 shows the blocking performance of channel grouping as a function of the group size G, with $m = G$ as $N \to \infty$ and compares the performance with respect to the case without grouping As the figure shows, grouping leads to significant reductions in blocking probability

This generalized group knockout principle forms the basis of the concept of multi-channel switching presented in this chapter The performance of the channel grouping can be further improved by increasing the maximum number of simultaneous arrivals, acceptable for a group (i e. using $m > G$) We define *Group expansion ratio* as the ratio of the number of output ports alloted to a particular group in a grouping network to the

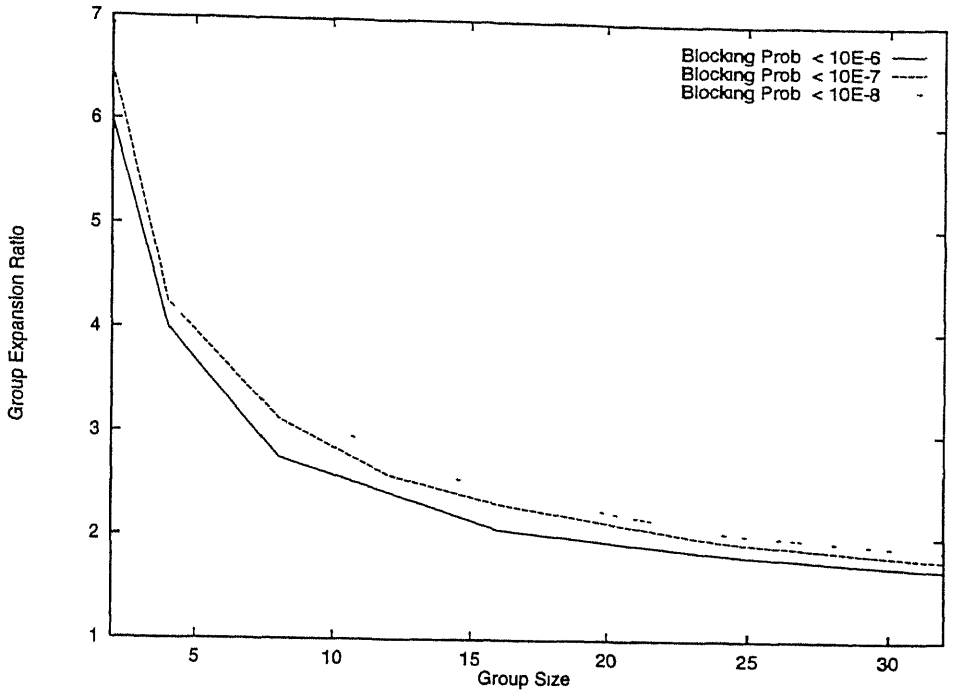


Figure 3 2 Variation of Group Expansion Ratio with group sizes

actual group size (Group Expansion Ratio $= m/G$) *Knockout efficiency* is defined as the reciprocal of the group expansion ratio required to satisfy a given blocking probability

The group expansion ratio as a function of the group size for various blocking probabilities using Eqn (3 5) is shown in the Figure 3 2

3.3 Alternatives for Channel Grouping

One of the first architectures with multichannel switching is by Pattavina [35] The switch proposed by Pattavina is internally nonblocking and is based on the Batcher-Banyan network It requires $O(\log^2 N)$ stages of interconnected 2×2 switching elements for N input ports and uses input buffering to resolve output contention and a three phase algorithm to allocate bandwidth

Another example of multichannel switching is by Cruz [7] who uses deflection routing in an omega network of $\log_2 N$ stages This networks is self-routing and requires minimal complexity in call set-up control but has non-zero blocking probability To achieve a reasonable cell-loss probability ($< 10^{-6}$), the switch utilization must be low The output of the switch is partitioned into channel groups as a function of input load This implies

that as the utilization of the switch changes, it is necessary to modify the partition to maintain the same performance level

The SXmin switch design also follows the group knockout principle [20] It uses an $N \times N$ Batcher sorter, followed by $\log_2 N - 1$ stages of sort-expander (SX) modules arranged in the form of a complete binary tree This allows a better group expansion ratio and achieves low blocking probabilities Each stage of the SX module expands the number of inputs by a factor determined by the group knockout principle, thus accepting the simultaneous arrival of multiple input packets destined to the output group corresponding to each SX module

In the following, we discuss some alternative designs to implement channel grouping

3.3.1 Crosspoint Architecture

This architecture [4, 5] proposes the use of a crossbar architecture for the switching of the cells It uses the group knockout principle in order to reduce the number of crosspoints There are $L \times G$ crosspoints meant for each group of size G The grouping tends to reduce the value of effective group expansion ratio, L, required for a particular blocking probability Each crosspoint consists of an address comparison circuitry, to detect whether the destination address of an input cell belongs to its group The switch consists of multiple stages of grouping networks, with progressively decreasing group size, to achieve the single-channel switching The crosspoint architecture suffers from scalability problems

3.3.2 Nonblocking Multichannel Switch using Flip Networks

This design [33] uses a combination of a running adder and a flip network as a nonblocking group network (NBGN) as shown in Figure 3 3 Its function is to separate the inputs into two groups in a nonblocking fashion Separation into more than two groups can be achieved by cascading a number of NBGNs, with each NBGN doing the partition based on a specified bit of the destination address For example, to separate the input cells into g groups, we need $\log_2 g$ cascaded NBGNs with the successive NBGNs partitioning based on sucessive bits of the group address The authors also suggest the use of a single NBGN recursively with a single cell buffer between successive recursions because the interstage

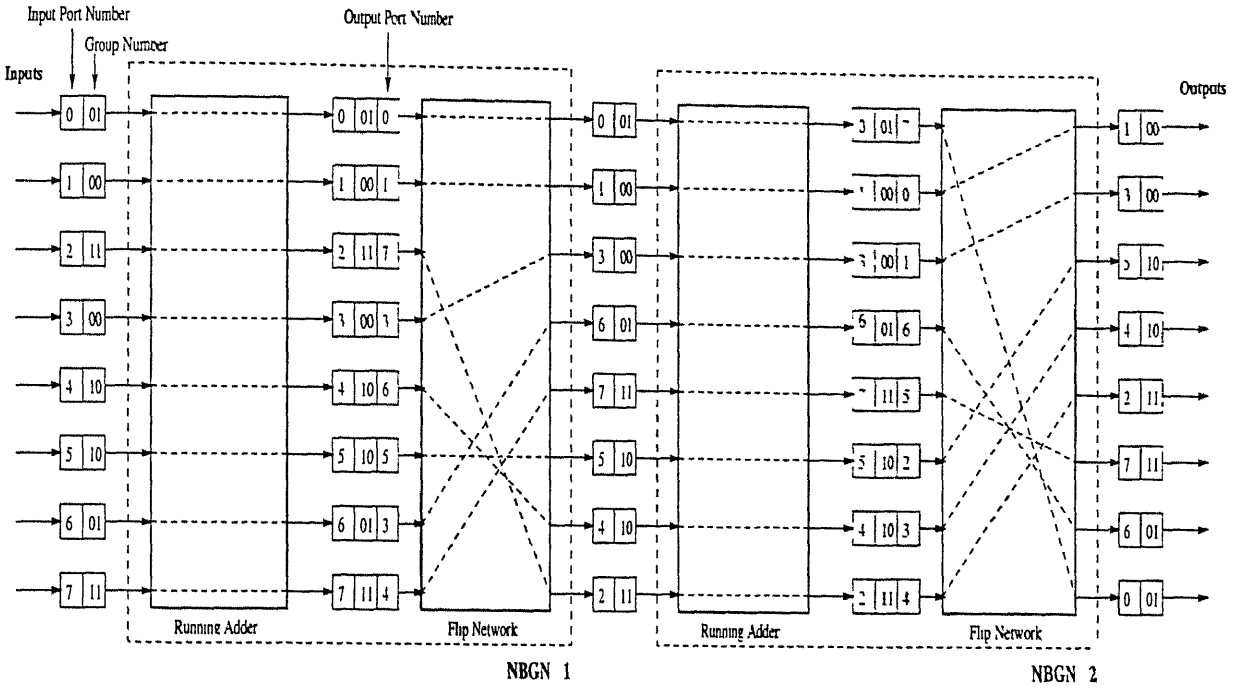


Figure 3 3 NonBlocking Group Network (NBGN)

connections are repetitive This combination, shown in Figure 3 4 is called recursive nonblocking group network (RNBGN) In order to use the same NBGN recursively, the propagation time of the cell through a particular stage of recursion must be reduced such that the recursions are completed in one cell time This can be achieved by transmitting data in parallel ([33] suggests 8 bit) An internally nonblocking multichannel switch can therefore be realized by a flip network of $\log_2 N$ stages This architecture is nonblocking in the multichannel sense and an arbitrary number of channel groups can be supported by controlling the number of recursions performed over the single NBGN This architecture suffers from severe scalability problems in terms of the size of the switch and that of the groups that can be supported To construct a multichannel switch with large N (and hence a potentially large number of channel groups), the use of a flip network with an expanded fan in ($k * N$) is recommended As cells arrive at the input, they are buffered and collected for k consecutive time slots Then each input line is distributed over k input channels in the kN-input RNBGN

The architecture has obvious limitations The recursive use of a single NBGN poses synchronization problems and suffers from speed limitations The expanded flip network increases the latency of the cell in the switch Moreover, if the architecture is used

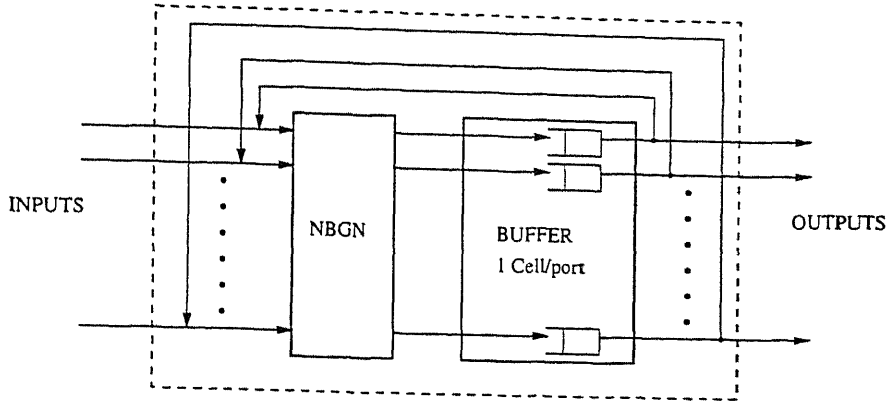


Figure 3 4 Recursive NonBlocking Group Network (RNBGN)

in the non-recursive mode, it does not offer any complexity improvements over other architectures The most severe limitation is due to the fact that the number of output ports assigned to a particular group is equal to the number of ports belonging to that group i e the group expansion ratio is 1 Thus the blocking performance of this switch is very poor even though the knockout efficiency is 1 (because of nonblocking nature in the multichannel sense) In order to achieve reasonable loss probabilities, the switch should be operated at very low loads

3.3.3 Grouping Network in Growable Switch Architecture

The growable ATM switch architecture proposed by Karol et al [8, 23] uses a Clos network configuration to achieve channel grouping (See Figure 3 5) The grouping part of the switch consists of two stages of switches The first stage consists of k, $n \times m$ input modules, followed by m, $k \times k$ switches in stage two Each second stage switch has one output per output group

The first stage consists of asymmetric modules achieving group expansion to improve the blocking performance of the switch These input modules also use a processor that uses a distributed algorithm to select the route for all the arriving coming in a time slot The algorithm selects a particular second stage switch through which the cell should be routed to its appropriate group, for each input cell The algorithm although distributed, operates in a sequential manner by dividing the full time slot into a number of minislots This number is equal to the number of groups During each minslot, each input switch finds the route for its cells destined to a particular group According to the algorithm

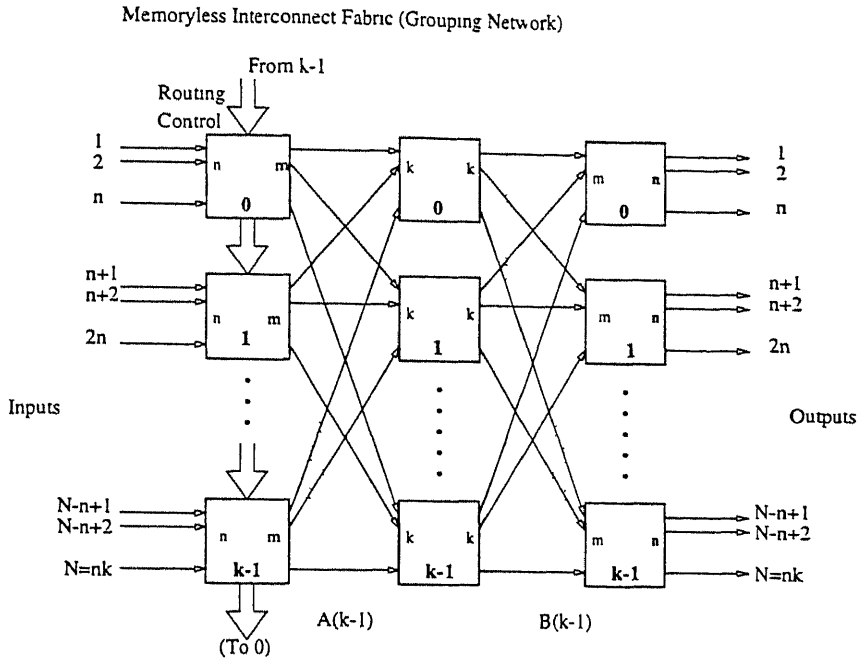


Figure 3 5 Growable Switching Architecture

suggested by Karol et al, in the first minislot, input module i tries to schedule its cells destined to output group i ($i = 0, \ , k-1$), where k is the number of groups Then in the h^{th} minislot ($h = 1, \ , k$), input module i attempts to schedule its cells destined to output module $((i+h-1) \bmod k)$ This scheduling uses only the local information - the status (i e busy or free) of its m outgoing paths (i e from the i^{th} input module), and the status of the m incoming paths to the $((i+h-1) \bmod k)^{\text{th}}$ output module The latter information is passed to the i^{th} input module from the $((i-1) \bmod k)^{\text{th}}$ input module It uses two vectors, A_i and $B_{(i+h-1) \ mod \ k}$, m binary elements each, to represent the status of the corresponding input and output module paths Note that each outgoing (and incoming) path corresponds to a particular second stage module. A cell can be routed from the i^{th} input module to the $((i+h-1) \bmod k)^{\text{th}}$ module only if there is an available path between the two stages through a common second stage module This can be easily found at each input locally by comparing the two bit vectors This path assignment process operates like a daisy-chain from one minislot to the next

Although the scheme is not optimal, it achieves very high knockout efficiencies The main drawback is due to the minislot concept and a complex routing algorithm

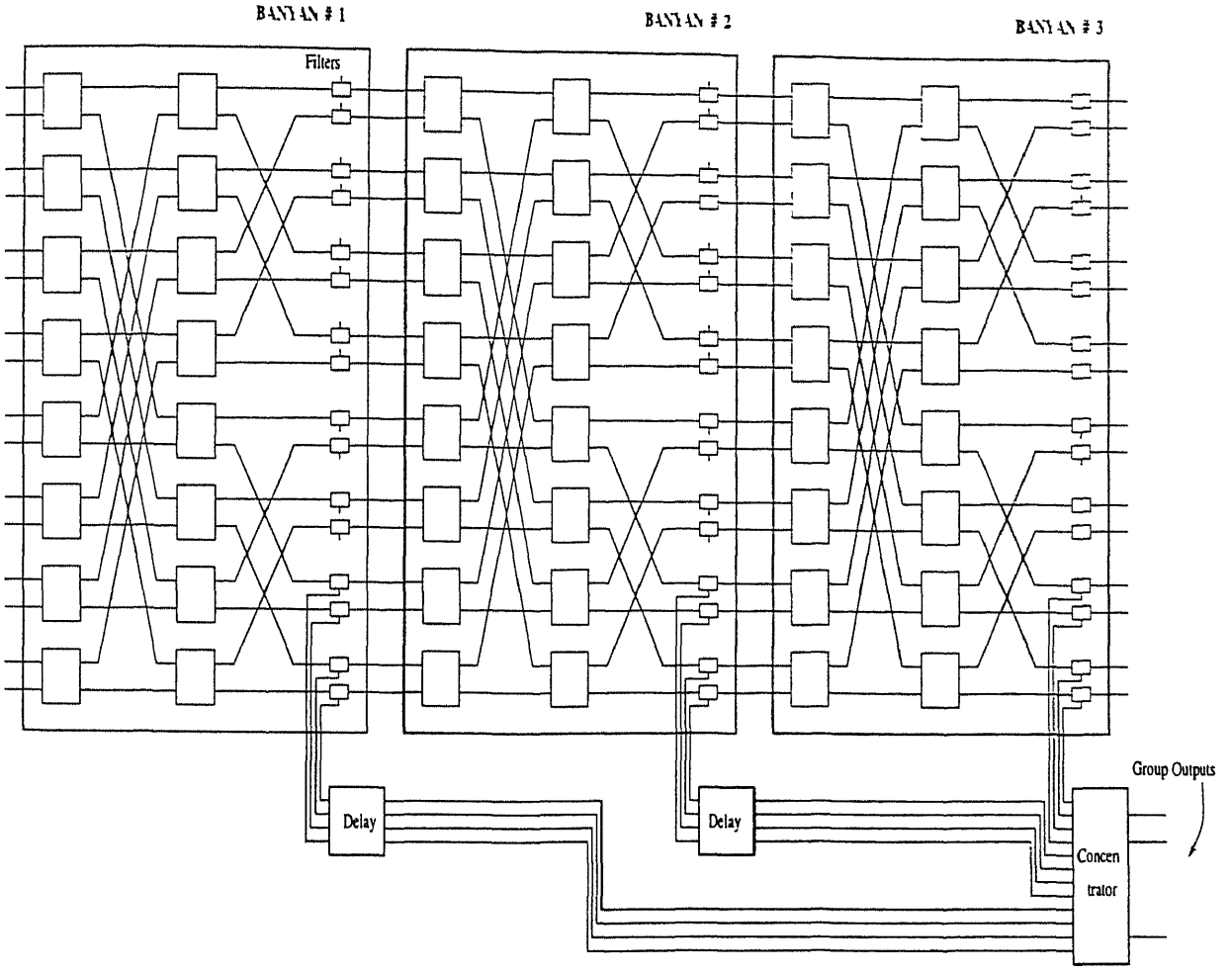


Figure 3 6 A 16 × 16 Tandem Banyan as a Grouping Network for Group Size = 4

3.3.4 Channel Grouping using Tandem Banyan Switching Fabrics

An alternative way to separate the cells destined for different groups is to use a tandem banyan switch fabric [42, 38]. This was explored in [12] In this design the use of shuffle network as a building block leads to various complexities in handling the mask bit We propose to use the modified data manipulator network topology (the modified data manipulator has been discussed in [42]) for the banyan networks The loss performance of this network is almost identical to that of the shuffle network [38] and the topology allows us to achieve the group switching with less number of stages Thus the switching elements corresponding to those in [12], with mask bit equal to zero are not required at all The switching elements of each of the banyan networks of this tandem banyan switch fabric will route according to the group address bits Thus if groups are formed from

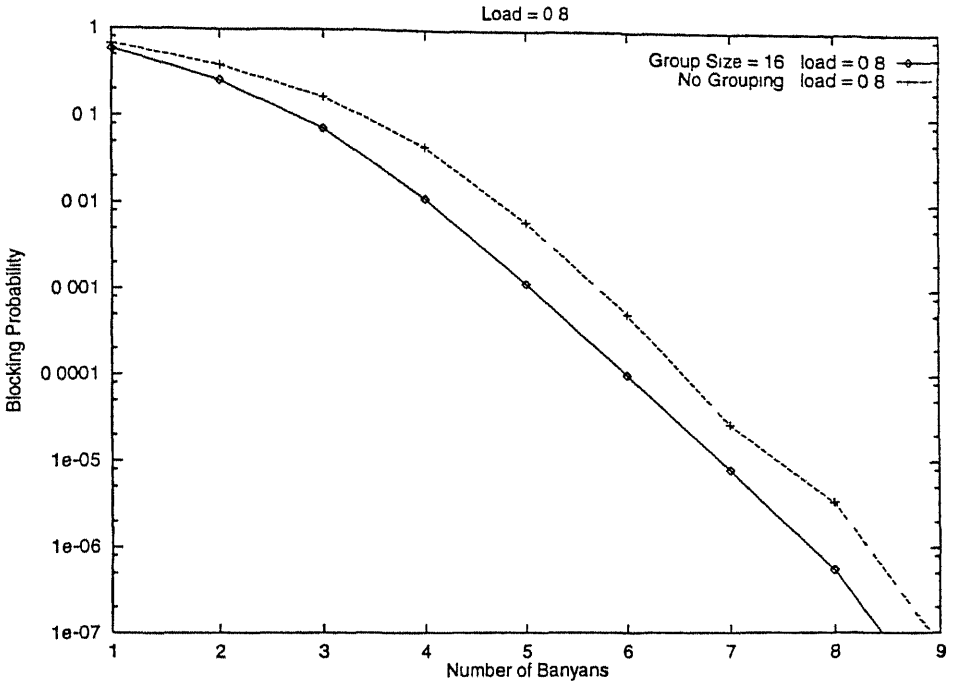


Figure 3 7 Performance of 1024 × 1024 Tandem Banyan as a Grouping Network

consecutive outputs, the leading bits of the individual destination addresses can be used as the routing bits for the banyans Each banyan will have $\log_2(N/G)$ stages of $N/2$, 2×2 switching elemants as shown in Figure 3 6 The outputs from the individual banyan will be grouped in sizes of G Each individual switching element follows the same algorithm as in the standard tandem banyan, i e it will route the cells to either upper or lower output depending on the bit corresponding to the stage in which the switching element is present, unless there is any conflict In case of a conflict, one of the two cells is routed properly and the other is misrouted and marked The marking is used to prevent the misrouted cell from affecting the routing of other cells in the later stages At the output of each banyan, filters are used to select the cells which are routed properly and feed them to the output buffer of that group The misrouted cells are passed to the following banyan Thus the load on the successive banyans reduces which has a positive effect on the blocking performance of the tandem banyan

The main advantage of using the tandem banyan as the grouping network is its simple structure and the self routing switching elements The number of banyans required will be a function of the acceptable blocking probability

The performance of the tandem banyan as a grouping network is shown in Figure 3 7

The graph shows considerable improvement in the blocking characteristics of the tandem banyan using channel grouping The main disadvantage of the tandem banyan is its poor knockout efficiency. Thus, in order to achieve a given blocking probability, the number of outputs per group required for tandem banyan is much more than that expected from the ideal in Eqn (3 5)

3.4 Conclusion

As we have seen in the previous sections, channel grouping can be incorporated in the design of ATM switches, leading to considerable reduction in the blocking probability Moreover as we will see in the chapter 5, the channel grouping along with shared buffering can be effectively utilized to construct large ATM switches

We also studied various architectures for multichannel switching. Although we realize the need for further research in this direction for a better architecture, among the various architectures described, tandem banyan seems to be a good choice Apart from its simplicity, the self-routing nature of the switching fabric renders it very suitable for constructing large switches Although it has a poor knockout efficiency and large group expansion ratio, a concentrator following the tandem banyan fabric may be used to improve it

Chapter 4

Shared Memory Switching

4.1 Shared Buffering

Output contention demands buffering to be provided somewhere in the switch in order to avoid excessive cell loss and to achieve good throughputs In a buffered switch, the cells which lose their chance in a particular cell time are buffered till a later time when they can be transmitted, rather than being dropped Thus the throughput of the switch can be increased Depending on the buffer location, we can classify switches into three kinds - input buffered, internally buffered and output buffered

Input Buffered Switches

The input buffered switch have buffers at each input They use some kind of feedback to know whether the cell at the head of the queue of an input port can go through in given time slot If it can not, it is buffered at the input and transmission is attempted again in the next time slot These switches have the advantage that the management of buffers is easy (essentially the input buffers are completely partitioned FIFO) and that each individual buffer is accessed at most two times during one cell time Thus they don't have stringent speed requirements On the other hand input queuing suffers from Head Of the Line (HOL) blocking [16 22, 41] This renders the switch non-work conserving, driving its throughput below its nominal value It has been shown that the maximum value of throughput decreases with N and reaches a limit rapidly as $N \to \infty$ Under the independent uniform

random traffic assumption, the limit is 0 586 and is reached very closely for N as small as 8 [41]

Internal Buffering

These switches have buffers in individual switching elements If there is contention in a switching element with more than one cell wanting to go to the same output all cells except one are buffered at the switching element Thus a cell would not have to contend for the part of the path it successfully managed to pass through Switches using buffered Benes [3] or buffered banyan networks are examples of internally buffered switches In these switches a small buffer is required at each switch element and no buffer sharing is possible, leading to poor buffer utilization Internal buffering may also cause sequencing violations in multipath switch fabrics

Output buffering

In this type of switches, the switch is capable of transferring more than one cell to a given output in one cell time Only one cell can be transmitted to the output line and the excess cells have to be buffered at the output Each output port has a dedicated buffer of its own Theoretically, these switches can achieve very high throughputs. In order to switch multiple cells to an output port during a single time slot, these switches use speedup of the internal fabric, or duplication of switching hardware elements to provide multiple paths from an input port to an output port

Output buffer architectures offer the advantage of simple control However a large buffer is necessary at each output port to keep the cell loss probability within acceptable limits. To reduce the buffer requirements, a buffer can be shared among many output ports, leading to *Shared Memory Switches* Some advantages of the shared memory switches are listed below

Statistical Advantage

The statistical sharing of buffers among many output ports results in a lower buffer requirements to achieve a given cell loss probability

Bandwidth Management

Shared memory queues are not limited to FIFO discipline and can incorporate

advanced queue management techniques such as per-VC queueing packet level discard, weighted round robin queuing and weighted fair queuing [11] It also offers greater price/performance benefits since it allows centralization of much of the implementations of the switch's advanced traffic management mechanisms [18]

Ease of supporting Priority and Multicasting

The most important feature of the shared memory architecture is that, they don't need to make physical copies of a multicast cell Instead they need to store only one copy of the original cell The same cell is read repeatedly by different output ports to achieve multicast functionality Priority can also be easily supported by maintaining different logical queues for different priority classes [24]

Keeping these advantages in mind, shared memory switches are the most efficient and cost-effective solution for small multicast ATM switches The applicability of the shared memory architecture to larger switches however, is limited due to technology constraints The rest of this chapter deals with the shared memory switches

4.2 Shared Memory Switches : Design Issues

4.2.1 Speed Constraints

The memory bandwidth should be sufficiently large to simultaneously accomodate all input and output traffic If N is the number of ports and V the port speed, then the memory bandwidth must be $2NV$ Given the limitations on the memory access speeds, there are two alternatives to realize a memory area for a shared buffer - registers [9], and RAM [24, 37] While the former is advantageous in terms of speed, with each register being accessed at the port speed only, the latter is advantageous in terms of integration density [37] We prefer to use the RAM because of its VLSI efficiency

The memory bandwidth required for the RAM is achieved by increasing the bus width [24, 37, 41] The incoming bit stream is first converted to a parallel data stream, and then the data is transferred in parallel in blocks The width of the bus depends on the

memory access time limitations and the size of the switch It should be noted that we cannot continue to increase the size of the switch to larger sizes by simply increasing the bus width because a limit will be reached at 384 bits, which is equal to the ATM cell payload length This limits the bandwidth capability of this class of switches to about 10 Gbps with the present technology [39]

In the architecture that we will propose in this chapter, the ATM cell payload stream is divided into eight substreams Each of the eight substream is converted into parallel and transferred to the memory in a single access We thus achieve an effective bus width of 384 bits

4.2.2 Buffer Sizing

The size of the buffer is determined by the specified dropping probability and on the extent of buffer sharing The required buffer size has to be estimated taking the expected traffic characteristics into consideration Quantitative estimates for the buffer size can be found in [10, 19].

4.2.3 Priority

Various media handled by an ATM switch will have different requirements of switching delay and cell loss rate. Therefore it is preferable to service a cell according to the service class to which the cell belongs Multiple priorities are generally handled by maintaining separate logical queues for each priority class and a deterministic rule determining the order of service among them [24]

4.2.4 Multicast Support

Multicast support is necessary in the ATM switches In some architectures, physical copies of the cell are made and stored into separate output queues In others only a single copy is stored and the copies are produced at the time of transmission and the original cell is removed after all the copies are made In this approach, detecting the time at which the memory can be released could be difficult because the memory can be released only after all the copies have been transmitted One way to overcome this

limitation is by giving highest priority to the multicast cells such that all the destination ports read the original cell from the memory in a single time slot, after which the memory can be released [24] This means that a separate queue needs to be maintained for the multicast traffic Maintaining a separate queue for the multicast traffic would mean that only one multicast cell would be released in a slot This affects the throughput of the switch and in the worst case, the throughput can drop to $2/N$ cells per port [24] Moreover, giving highest priority to the multicast traffic may not always be acceptable In the architecture that we propose in this chapter, we overcome these limitation through a counter based scheme

4.3 Proposed Architecture for the Shared Memory Switch

The proposed architecture for an $m \times n$ shared memory switch has the following functional modules (See Figure 4 1)

1 Memory Modules

2 Controller

3 Routing Table

4 Destination Counter Module

5. Output Module

Memory Modules

The header part of the incoming cell stream is forwarded to the controller while the payload portion is fed to the memory module The payload stream is further divided into eight substreams Thus each substream now carries 48 bits during one cell time These substreams are fed to eight different memory modules, all of which perform identical operations, concurrently

Each memory module has an internal structure as shown in Figure 4 2 The input lines are terminated into Serial to Parallel Convertors (SPCs), which convert the

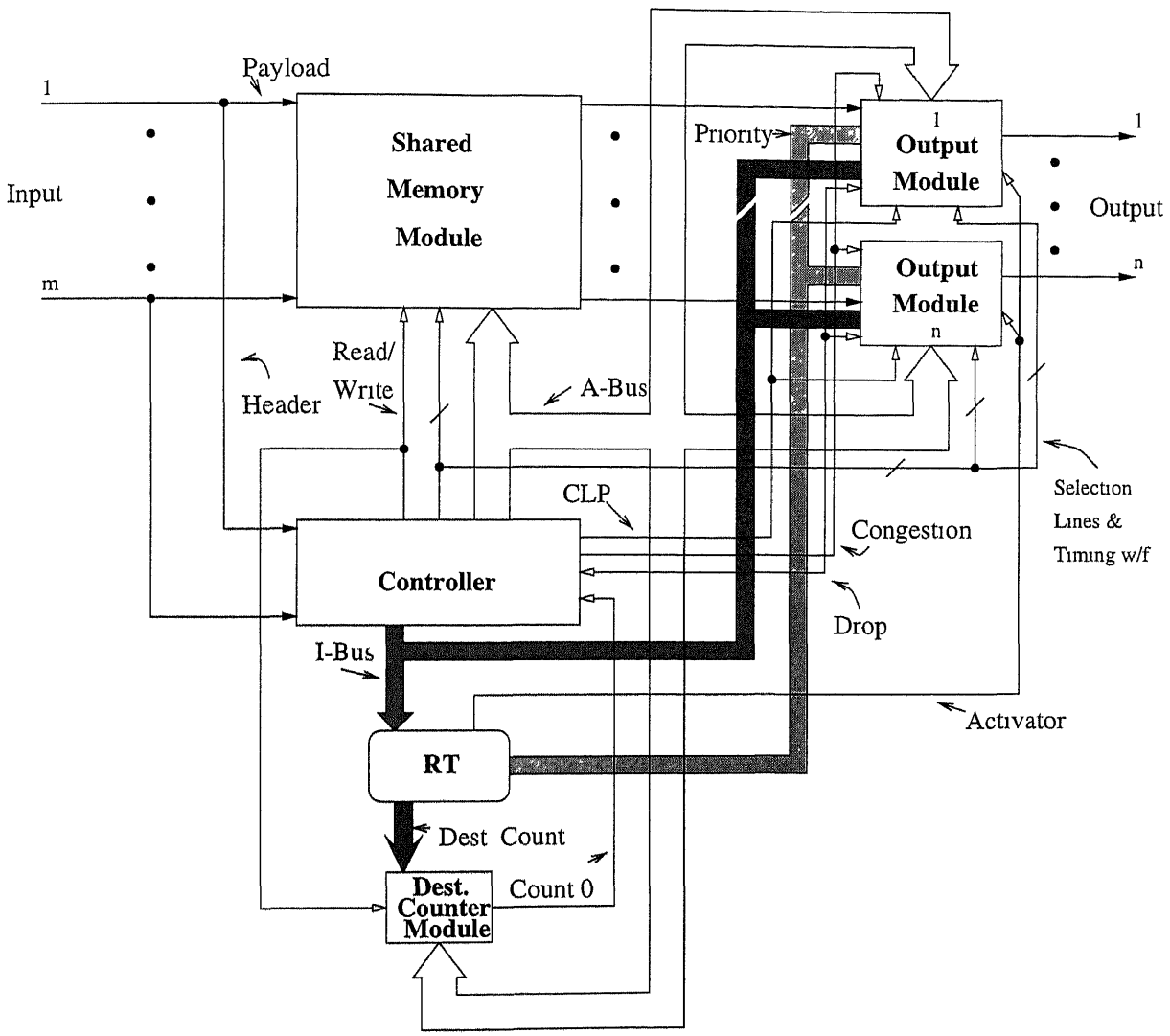


Figure 4 1 Shared Memory Switch Architecture

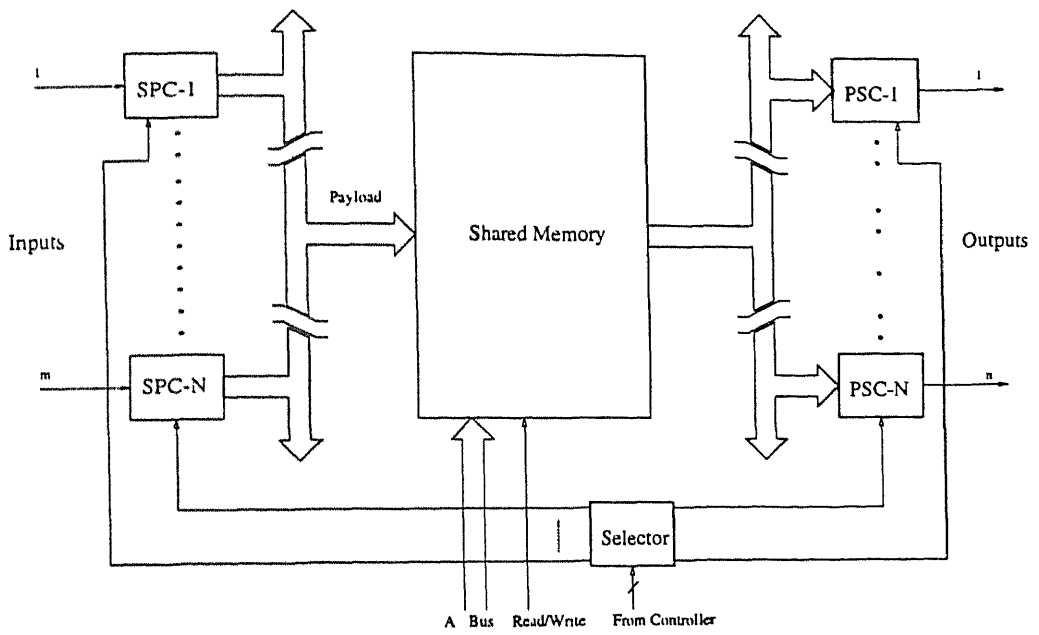


Figure 4 2 Memory Module

incoming bit stream into a 48 bit wide parallel word These SPCs are interfaced to the common buffer memory through a single 48 bit wide bus The SPCs access the memory in a round-robin fashion

On the output side, the memory is interfaced to Parallel to serial convertors (PSCs), which when activated, convert the parallel data on the bus into a serial stream and transfers it to the corresponding output module The controller provides the read/write control signals and the address for the memory It also generates the selection signals required to enable the SPCs and PSCs

Controller

The controller, shown in Figure 4 3, is the heart of the switch It generates all the timing waveforms required for the proper functioning of the switch It also determines and times the order in which different inputs and outputs access the memory In order to allow all the input and output ports to access the memory in a round-robin fashion, it divides one cell time into a number of slots This number is atleast equal to the sum of the number of input and output ports Thus only one device is allowed to access the memory at a time

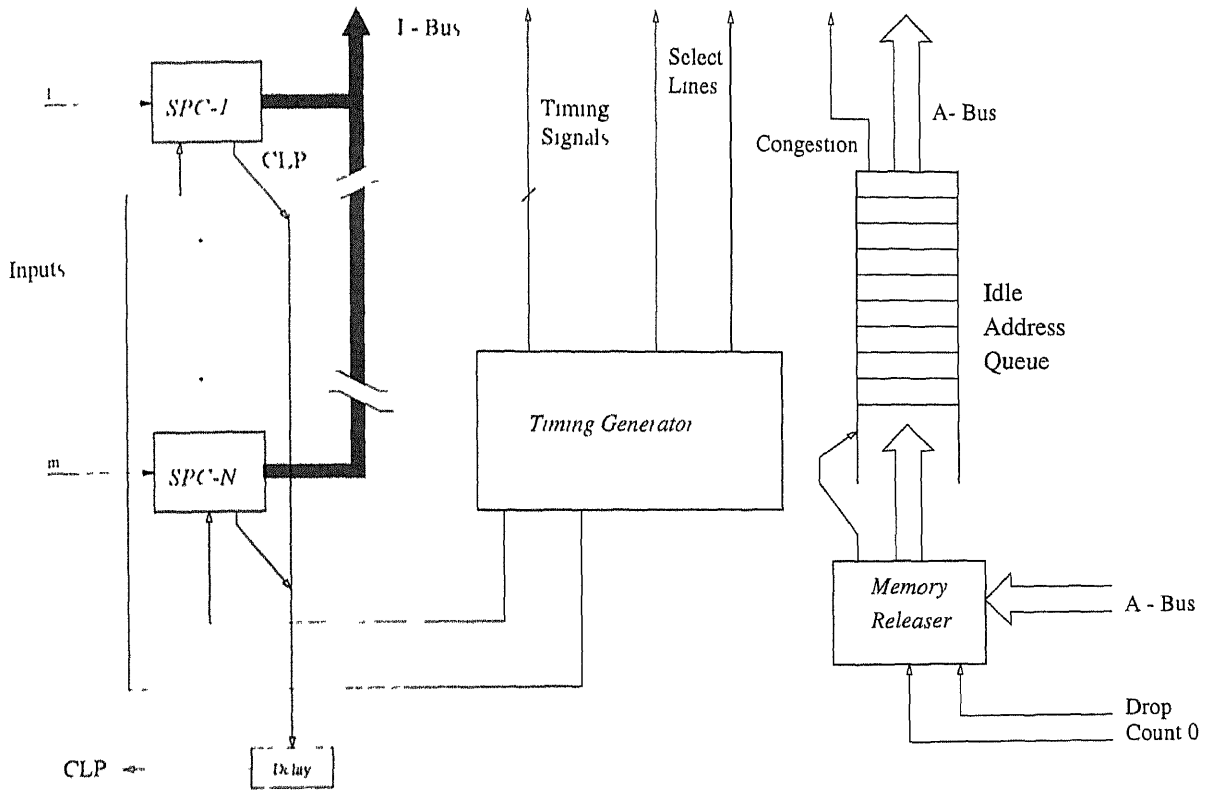


Figure 4 3 Controller

The header stream coming from the input line is first converted into a parallel form using SPCs All the SPCs are terminated into a single bus called the Index Bus (I-Bus). The order and the time when the SPCs access this bus are determined by the controller timing generator

When selected, an SPC places the input VCI/VPI on the index bus It also gives an indication of the input CLP on the CLP line after a delay of one slot The contents of the header index the routing table

The controller also has a FIFO memory, which stores the addresses of all the memory locations which are unused at any time Whenever a new cell arrives, a new address is fetched from this FIFO and is placed on the Address Bus (A-Bus) On the other hand, whenever a memory becomes free, the address of that location is appended to the queue. The Memory Releaser accomplishes the function of releasing the memory. Due to the multicast function, this process of releasing the memory, is not straight forward and will be discussed later

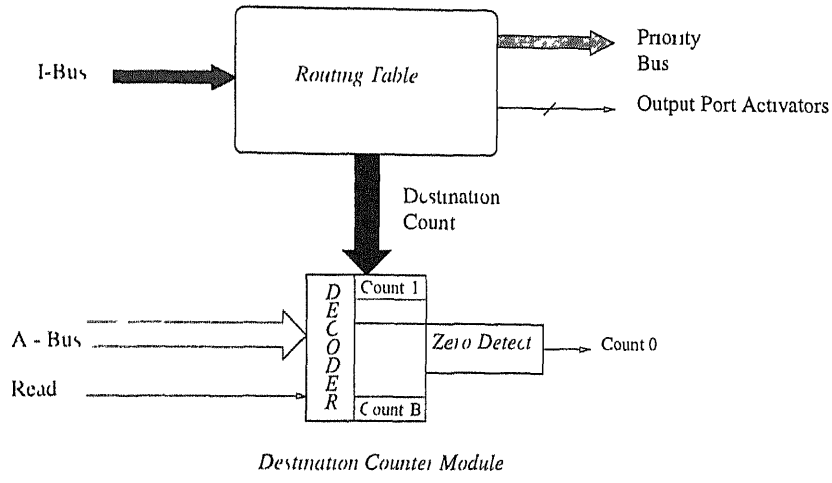


Figure 4 4 Routing Table and Destination Counter Module

Routing Table

The routing table (RT) (Figure 4 4) contains a bit pattern corresponding to the destinations of the particular connection For an n output switch, it stores an n bit vector, indicating the output port number(s) to which a cell belonging to the particular connection needs to be forwarded Apart from the routing information, it contains additional information such as priority required by the scheduler It also keeps a count of the number of destinations for the connection The routing table is accessed using a CAM (Content Addressable Memory)

The routing table is maintained by the call processing layer processes The contents of RT are determined during the connection setup time and may be changed later in case of renegotiation of connection parameters The header contents available on the index bus serve as an index to the routing table

The output destination bit map is used to activate the corresponding output ports The priority information is put on the Priority Bus (P-Bus)

Destination Counter Module

This module (Figure 4 4) consists of a number of down counters, the number being equal to the storage capacity of the shared buffer Each entry in the destination counter module contains the destination count associated with the cell stored in the memory at the same address. The destination count signifies the number of

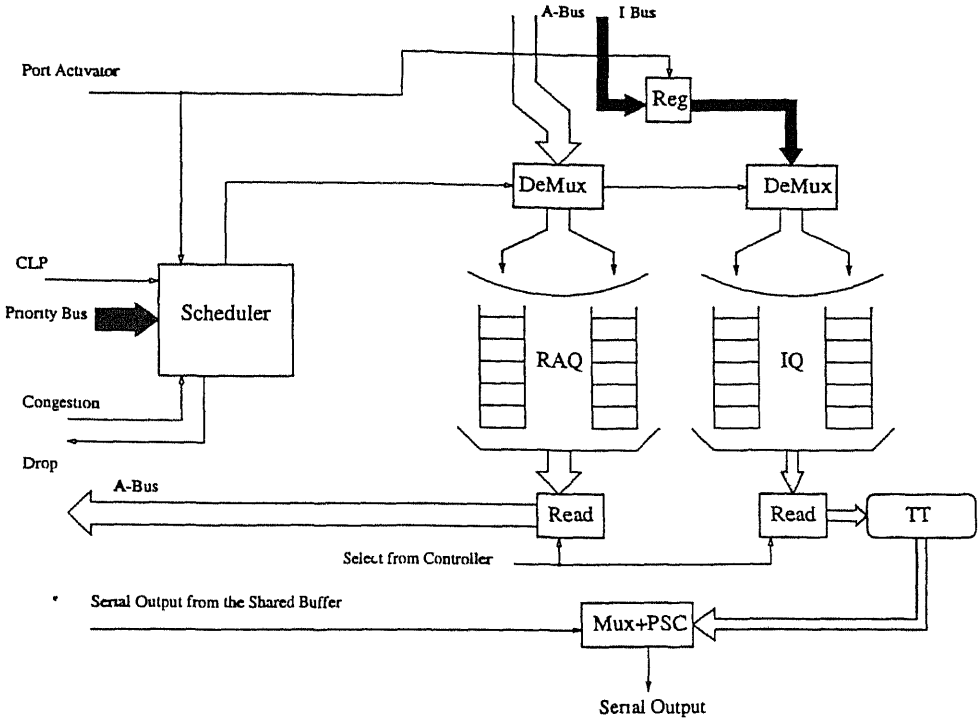


Figure 4 5 Output Module

memory module Every time an address in the shared buffer is selected for read operation, the corresponding destination count is decremented by 1 If the destination count goes to zero, an indication is provided to the controller module to release the associated memory This can then be allocated to store incoming cells This destination count approach overcomes the throughput limitations of [24] without much complexity overhead

Output module

The output module shown in Figure 4 5 contains a table called Translator Table (TT) which contains the translated header information (outgoing VCI/VPI values) for the active connections passing through the associated output port

The output module also contains a scheduler, which when activated, determines a queue pair from among several, depending on the information on the P-Bus and CLP of the cell, as determined by the scheduling policies The scheduler is informed of a possible congestion condition in the switch by the controller, in which case the scheduler drops unimportant cells and asserts a drop indication for the memory releaser in the controller This drop indication signal is also monitored by each

output ports The cell is dropped and the associated memory is released if any of the output module asserts the drop indication

The output module has a number of queue pairs, corresponding to different classes the switch needs to support Each queue pair consists of a Read Address Queue (RAQ), which is used to store the address of the memory location containing a cell for the output port, and an Index Queue (IQ) containing an index to the translator table associated with the output port

Whenever the header contents are put onto the index bus for accessing the routing table, the index is stored in a register in the output module If the cell is accepted by the scheduler, this index is saved in the IQ This index is later used to index the translator table, to get the translated VCI/VPI number of the corresponding outgoing cell The address at which the cell is written, as determined by the contents of the A-Bus, are appended to the RAQ

When activated by the controller, the output module reads the contents of the RAQ of the highest priority non-empty queue pair and puts it on the A-Bus This address is used to access the memory at which the cell destined to the output port is stored In parallel to this operation, the index from the corresponding entry in the Index-Queue (IQ) is used to access the Translator Table (TT) to get the translated header information for the cell The output module receives the serial bit stream correponding to the cell payload and adds the corresponding output VCI/VPI information before transmitting it to the output port

4.3.1 Working of the Proposed Switch

As already mentioned, controller divides the total cell time into a number of slots As there are a number of subblocks such as SPCs and PSCs using the memory the controller is responsible for coordinating these accesses In order to time share the memory among m inputs and n outputs for an $m \times n$ switch, there has to be atleast $m + n$ slots in a single cell time

Every time a new cell comes to the switch, it has to be written to the memory The processing of a new cell arriving to the switch can be partitioned into two independent

phases In the first phase the header of the cell is decoded to get the intraswitch routing information for the cell The second phase is responsible for accessing a free memory location and storing the cell in it Every time a cell is written to the memory, all the associated destination port modules are informed to add this address to their appropriate RAQ, depending on the priority of the connection

In order to manage these two processes efficiently, the processing of a new cell is done in two consecutive slots In the first slot, the controller decodes the header and places the header contents on the I-Bus and CLP line In parallel to this operation, the controller accesses the idle address queue (IAQ) to get the address of an unused memory location Apart from the routing table, for which it acts as an index, the I-Bus contents are read by all output port modules The output port modules store this index temporarily Depending on the output of the table, appropriate port modules are activated in the next slot to use this temporarily stored index, as explained earlier

In the following slot, the address read from the IAQ is put on the A-Bus. The output of the routing table (as indexed in the previous slot), is used to activate the appropriate output port modules (which belong to the group of destination ports of the cell whose header was processed in the previous slot) The activated output port modules examine the queue status, priority informaion and the congestion indication to decide whether the cell has to be accepted or not In the later case a drop indication is provided to the controller to release the memory If the cell is accepted, the scheduler uses the priority information and the contents on the CLP line to determine the class to which the cell belongs to and writes the contents of the A-Bus to the corresponding RAQ In parallel to this operation, the output module writes the index stored in the previous slot to the Index Queue (IQ) at a position corresponding to the address of the cell stored in the RAQ The memory is accessed for the address determined by the A-Bus contents and an appropriate input SPC is activated to transfer the cell payload to the memory location

Thus the complete processing of the input cell takes two time slots As there is no overlap in the resources required for these two slots, the processing of input cells can be pipelined to meet the speed constraints Thus a new cell can be taken up for table access in the slot in which the previous cell is written to the memory

The processing of the output cells is also done in a similar manner Each output

port gets its turn one by one as determined by the controller When any output port module is activated for memory read it checks its queues and accesses the RAQ of the highest priority non-empty service class In parallel to this operation the corresponding contents of IQ are used to access the translator table (TT) to get the translated header information The contents of RAQ are put on the A -Bus in the next slot, which are used as an index to the memory The cell is read from this address and the output is stored in the corresponding PSC The PSC converts this parallel data into serial form and passes it to the corresponding output port module In the output port module, the translated header information from the TT are appended to the cell before its exit from the switch The destination counter decrements the count associated with the memory, every time it is read and gives an indication to the controller to release the memory, when the count decrements to zero

4.4 Performance of the proposed Shared Memory Architecture

The proposed shared memory architecture differs from a comletely shared memory architecture, because the logical size of any output queue is restricted by the depth of the RAQ and IQ at the output ports This type of memory sharing is called SMXQ (Sharing with MaXimum Queue constraint) [10, 19, 21] This scheme of memory sharing achieves a buffer utilization close to that achieved by the complete sharing scheme but offers better performance for unbalanced traffic conditions as the maximum queue constraint puts limit on the maximum buffer share obtainable by any output port and hence sudden overload on a single heavily loaded output port does not deprive lightly loaded output of the buffers The analysis of such a scheme of buffer allocation is quite involved and is not included here However, according to the results shown in [10] an output queue size equal to half of the total buffer size gives the best performance under unbalanced traffic

The buffer size depends on the traffic conditions If all inputs generate narrowband CBR (continuous bit rate) traffic, cells come to the switch randomly because of multiplexing cells from a lot of inputs during a short term (i e , several milliseconds) If all the inputs generate wideband CBR traffic, cells come periodically and the required buffer

is smaller than that under uniform traffic conditions For example, for a 32×32 switch handling CBR traffic, a thousand cell shared buffer will keep the dropping probability below 10^{-9} [24] The buffer requirements for bursty VBR sources are higher.

Chapter 5

Designing Large Multicast ATM Switches

5.1 Introduction

We have seen in chapter 1 that when the number of ports is large a single large switch is cheaper than a network of small switches Thus scalability of a switch architecture to large number of ports is an important issue

A number of architectures to build large ATM switches have been described in the literature The switch being designed at Washington University [3] uses a buffered Benes topology with individual shared buffered switching elements Instead of following the looping algorithm described in [17], a random routing algorithm is used in the first half of the Benes network and a self routing algorithm like that used in the Banyan network is used in the second half This increases the blocking probability in the switch inspite of the use of a nonblocking Benes topology Moreover, as each switching element is required to have internal buffers, the total number of buffers required in the switch is very large A most severe drawback of this architecture is the sequencing violations by the switching network, requiring elaborate resequencing mechanisms Multicast is supported using time-based multicast scheme

Batcher-Banyan based switch architectures like Sunshine [13] and SCOQ [6] are limited in the realizable size because the complexity of the Batcher sorter increases as $O(\log^2 N)$ Also, the trap networks that are required are difficult to realize The modular

architecture proposed by Lee [28] does not require large Batcher modules, but the overall complexity of the architecture makes it impractical due to a large number of Batcher sorters, Banyan networks and binary trees

The growable architecture of [8, 23] uses a Clos network topology to realize of large but switch has limited applicability due to its complex routing algorithm.

In the next section, we propose a new method to construct large multicast ATM switches The architecture is inherently scalable in nature and can be extended to large sizes like 1024×1024

5.2 New approach for building large multicast ATM switches

As we have seen in the previous chapters, the space-division switches are inherently scalable in nature, but need special circuitry for effective support of multicast functionality On the other hand, the shared memory switches can support multicast easily but are not scalable in nature Although the modifications in the design of copy network suggested in chapter 2, make practical implementation of the copy networks possible, the switching network speedup required in the scheme can become excessive for large switches This may render them unsuitable for use in large switches On the other hand, shared memory switches can not be extended to more than about 32 ports with the current technology, in view of the memory speed constraints This leads to a dilemma, as to which architecture to select for building a large multicast ATM switch To solve this dilemma we propose a hybrid approach in which the multicast functionality is achieved in two stages. The first stage is a space-division fabric that is called the space-division grouping network and the second stage is the output module which is a shared memory switch Very little buffering is provided in the space-division stage Almost all the buffering is done in the second stage and the buffer is shared among a group of ports

We make use of the concepts of channel grouping and the shared memory switching developed in the previous chapters Channel grouping greatly reduces the blocking probability in a space-division switch The output stage consists of shared memory switches, which do the intra-group switching In addition to reducing the blocking probability,

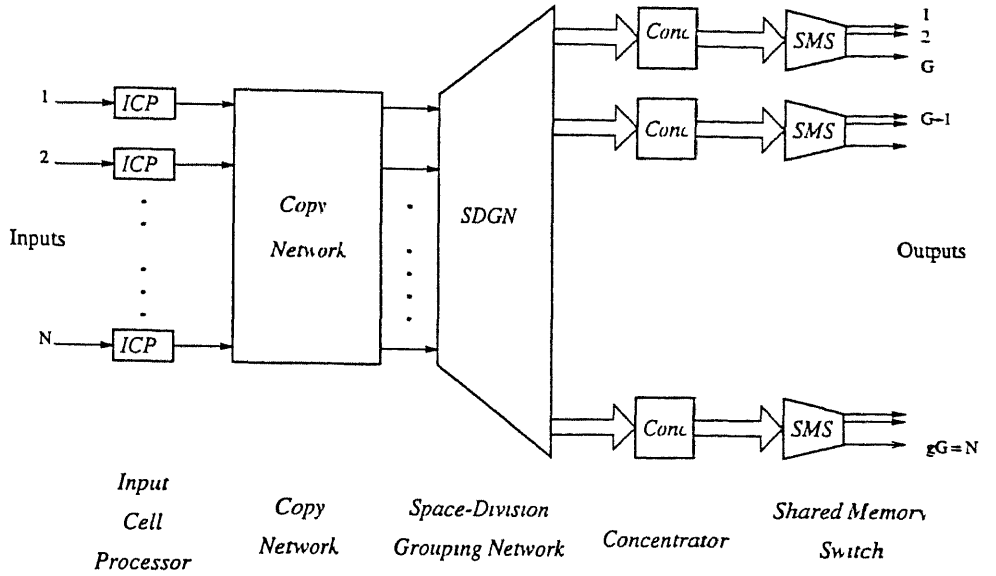


Figure 5 1 Block Diagram of the proposed architecture for constructing large multicast ATM switches

grouping also aids in multicast support In the architecture that we propose the multicast function is also done in two stages In the first stage, only one copy is made for all the destination ports, belonging to a group This not only reduces the load, leading to further reduction of blocking for multicast traffic, but also relaxes the speedup requirements in the space-division switch because in space-division stage we need to address the outputs on the basis of group numbers instead of output port numbers The second stage of the multicasting is done in the shared buffer output modules

The output modules are asymmetric with m input and G output ports This asymmetry is required to have a good group exapnsion ratio in the space-division grouping network to reduce the blocking probability

Since the group expansion ratio required for a certain blocking probability does not change much with the switch size, this architecture is highly scalable Thus arbitrarily large switches can be built using the same output modules. The block diagram of such a large switch is shown in Figure 5 1 and the details are explained below

The Input Cell Processors (ICPs) get the input cells and access the Routing Table (RT) to get the routing information (essentially a bit map corresponding to the output groups) for the cell This information along with the number of copy information is appended to the input cell as copy network header The ICP gives the cell to the copy

network The detailed architecture of the copy network was presented in the chapter 2 The output of the copy network is fed to the grouping network A tandem banyan switch fabric is used as a grouping network in this architecture, although any other suitable grouping network will serve the purpose The outputs of the tandem banyan fabric are divided into groups The number of outputs of the tandem banyan fabric for a particular output port group depend upon the number of banyan stages employed in the tandem banyan network The output of the tandem banyan network is passed through the concentrator, which reduces the number of outputs for a particular group The output of the concentrator is fed to the shared memory switch modules, one each for individual groups The architecture of these shared memory modules was explained in chapter 4 This module also performs header translation

In the next section, we will see an example design of a 1024×1024 switch, using the proposed architecture

5.3 Design of a 1024×1024 switch

The main issues when designing a switch of a given size are related to the performance aspects For the proposed switch, the tolerable blocking probability, overflow probability, delay etc are the main performance issues Once the base architecture is selected, the major task is in the tailoring of various parameters to meet the performance guarantees In the following, we list the performance measures and factors that determine them in the proposed architecture -

- Connection Blocking Sizes of the tables at the ICP, in the shared memory switching module and of the translator table at the output ports
- Blocking Probability Number of banyans in the tandem banyan grouping network, size of the concentrator at the output of the tandem banyan switch fabric, the number of buffers in the shared buffer in the output shared memory switches and the depth of the output read address queue (RAQ) and index queue (IQ)
- Overflow Probability The Buffer size required in the shared buffering stage of the copy network

- Delay Buffer sizes in the shared buffering stage at the output of copy network and in the shared memory switching modules at the output and sizes of the IQ and RAQ at the output ports

We assume that limiting the number of simultaneous virtual circuits (VCs) per port to 8000 would be sufficient to maintain a connection blocking probability due to table deficiency lower than that due to bandwidth limitation This choice is justified because even for the worst case of very low bandwidth applications of the order or few 10s of kilobits of bandwidth requirement, 8000 is close to the upper limit on the number of simultaneous VCs that can be supported on a 155 Mbps link

These tables are maintained by the call processing and control layer processes Once a connection request is accepted and a path through the network is established, an entry for the VC is created in the routing table (RT) of the corresponding ICP Also, depending upon the destinations, entries are created in the tables in the output shared memory switches Specifically, an entry is created in the routing table of each group to which any destination for the connection belongs to along with one entry in the translation table (TT) of each destination output port

Another important design issue when designing a switch using the proposed architecture is to decide on the appropriate size of the output modules Large input/output port size is desirable in terms of buffer sharing, as the cell loss rate tends to decrease for a given buffer size A shared buffer switch which has huge input/output port size, however, is difficult to implement due to physical limitations, such as memory speed, pin count limitation, power dissipation and so on [37] Even from the reliability point of view, it is better to have a large switch constructed from small switch modules, to avoid the danger of a single point of failure [11] Therefore the size of the switch is determined taking into account this tradeoff

In order to evaluate the sharing effect, we use the notion of buffer reduction ratio [10], defined as the ratio of following two buffer sizes the buffer size per port required for a shared buffer switch and that for an output buffered switch, so that both maintain the same level of cell loss rate under the same traffic load Because of the sharing effect. the buffer size required for the shared buffer switch is less than that for an output buffered switch, and therefore the buffer reduction ratio is generally a positive number less than

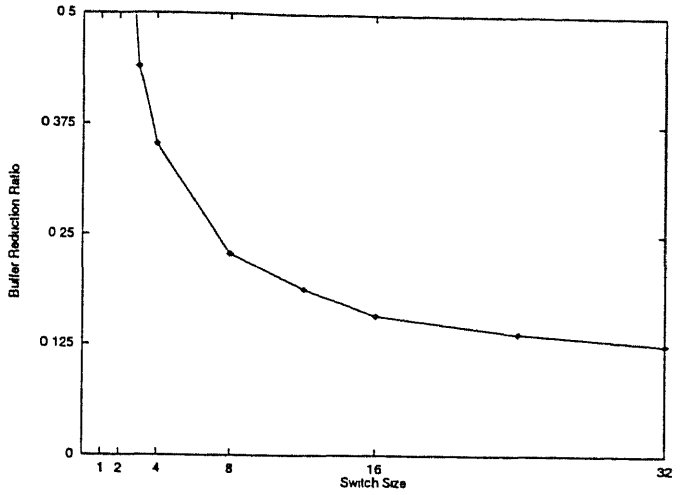


Figure 5 2 Variation of Buffer Reduction Ratio with the size of the switch for random traffic with load 0 9 and cell loss requirement $< 10^{-9}$

1 Figure 5 2 shows the relation between the buffer reduction rate and the port size It is clear that the buffer reduction ratio is decreased as the port size increases However the slope of the curve becomes small and small as the size increases. For instance, the buffer reduction ratios at $N = 4, 8, 16$ are 35 3%, 22 9% and 15 8% respectively This means that the sharing effect is improved by 12 4% when the size is increased from 4 to 8, but only 7 1% improvement is obtained when N is increased from 8 to 16

In the following, we use 16 as the size of the output shared buffer module A shared memory switch of this size can be realized with the present technology Moreover, the slope of the buffer-reduction ratio curve is almost flat beyond the size of 16

The blocking probability in a 1024×1024 tandem banyan with a group size of 16 for different concentrations is shown in Figure 5 3 It shows that 8 banyans with a concentrator output size of 35 are sufficient to keep the blocking probability in the grouping network below 10^{-6}

The size of the shared buffer in the output module depends on the kind of traffic expected in the switch In [10] it has been shown that for an 8×8 switch, a buffer size of 1500 cells is sufficient to have a dropping probability of less than 10^{-9} even for a bursty traffic This works out to about 188 cells/port for an 8×8 switch The buffer size for a 35×16 switch can be estimated as 130 cells/port using the buffer reduction ratio curve shown in Figure 5 2 This amounts to a total buffer requirement of 2100 cells/output module. To keep a margin of safety, we suggest a buffer of 2500 cells/output module

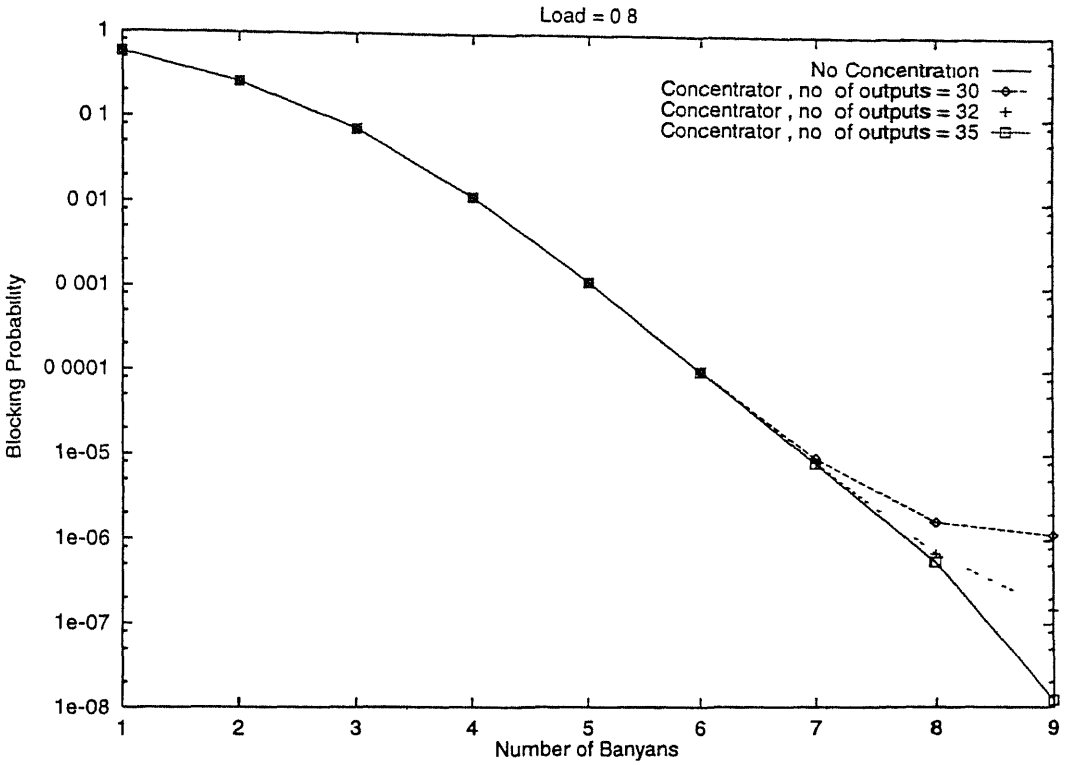


Figure 5 3 Performance of the 1024 × 1024 Tandem Banyan Grouping Network with group size of 16

The cycle time of the buffer memory will be of the order of 54 nanoseconds (35 + 16 memory accesses in one cell time of 2 73 microseconds) Keeping a safety margin, the cycle time will not be less than 45 nanoseconds

Due to large number of ports, even a single buffer per port for the shared buffer stage of the copy network is enough to achieve negligibly low overflow probabilities The overall delay characteristics of the switch is almost solely determined by that of the output buffer The latter depends on the traffic characteristics The output buffers, effectively act as a single point of buffering in the switch This allows use of complex queue management techniques

The performance of the switch can be improved further by intelligent grouping of the outputs such that most of the destinations belonging to a multicast group are grouped together in a single output group This tends to reduce the load on the space-division grouping network resulting in improved cell dropping characteristics

5.4 Conclusion

We conclude this chapter giving a rough estimate for the number of chips required for the practical realization for a 1024×1024 switch using the proposed architecture The assumptions for these estimates are in line with similar calculations done in [14 47 48]

The input cell processor can be accomodated in a single chip This amounts to 1 chip/port for ICP

The copy network consists of a 1024×1024 running adder network, followed by 1024 group splitting units and address encoders (GS-AE) This is followed by a 2048×2048 concentrator consisting of running adder network and reverse banyan network, both being 2048×2048 in size The concentrator feeds the distributed shared buffering stage, with a total 1024 single cell buffer being required The shared buffers feed the cells to the 1024×1024 RBN

The running adder network and the group splitting - addrress encoder unit can be combined into one and assuming 128 I/O pins per chip, a total of 16 chips are required for the realization of running adder and group splitter A 2048×2048 concentrator can be built using 32×32 modules, requiring a total of 128 chips The single cell buffers can be realized in the last stage of the concentrator itself Following the same approach, 64 chips will be required for the realization of RBN

The space-division grouping network consists of a number of banyan networks, connected in tandem Each banyan consists of 6 stages of 512, 2×2 switching elements Again assuming 128 I/O pins per chip, a total of 16 chips are required per banyan This amounts to a total of 128 chips for 8 banyans in tandem

The space-division grouping network is followed by a concentrator Assuming that a 128×35 concentrator can be realized in a single chip, the total chips required for the concentrators are 64

The Output shared memory module consists of memory modules, controller, output modules, routing table, translation tables and destination counter module Eight chips, each with a storage capacity of 2500, 48 bit-words, are required for the memory module in order to receive the eight input substreams One chip each is required for controller, routing table and destination counter module respectively In addition, the output modules and the associated translation tables require 1 chip each, per output port Thus the

total number of chips required for each output shared buffer module is 43

Thus the total chip count required can be estimated to be about 4176 which amounts to about 4 chips/port

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we looked upon various issues concerning the design of multicast ATM switches Out of the two popular approaches for constructing the ATM switches, namely the space-division and shared-memory switches, we studied the problems with the existing architectures for supporting multicast in space-division switches and proposed various schemes to overcome them Specifically, we developed the Memory-Optimized Lee Multicast Switch architecture, which minimizes the memory requirements for maintaining the connection control information We also carried out a comparative analysis of the memory requirements in this architecture with respect to the existing architectures and conclusively proved that our architecture offers considerable improvement over the existing ones Further to this, we looked at various mechanisms for reducing the problem of overflow in the space-division multicast switches Out of the various mechanisms studied, we found that the one employing shared output buffering turns out to be the best We proposed an architecture combining the principles of memory requirement reduction and shared buffering and discussed the detailed functioning of the same

Next, we introduced the principle of channel grouping in space-division switches and suggested a way to implement it We showed that channel grouping leads to considerable improvement in the performance of the space-division switches

Next, we looked into the shared-memory switches and developed an architecture for constructing small shared-memory switches incorporating multicast functionality We

found that the shared-memory switches are the preferred choice for constructing small multicast ATM switches

We concluded the thesis by suggesting a method to combine the space-division and the shared-memory approaches to realize large scalable multicast ATM switches. We also presented a rough chip count estimate for an example switch, along with dimensioning of various parameters

We have shown conclusively that the architectures proposed by us for implementing multicast ATM switches are more efficient both in terms of performance and VLSI realization, as compared to the existing architectures Although the VLSI implementation of the proposed architecture can begin around the existing ATM chip sets, we feel that newer functional chips will be required to be designed to implement our architecture.

6.2 Future Work

The area of ATM switching has been attracting a lot of research efforts in the recent past The scope of this field is immensely vast However, we have addressed a smaller area viz Multicast support in ATM Switches in this thesis Inspite of our efforts to look at all the issues, we feel that further studies are required in the following directions

Although the tandem banyan discussed in chapter 5 was selected as the best available choice for realization of the grouping network, we feel that there is a scope of further improvement

The analysis of the sharing with maximum queue constraint (SMXQ) strategy as used in our architecture has not carried out One can analyze the scheme to come up with optimum values of the output queue sizes for different buffer sizes to achieve desired dropping probabilities Moreover, all the simulations were based on uniform traffic model The results need to be extended for bursty traffic model

A considerable effort is required for VLSI realization of the architectures proposed by us and to develop a functional prototype of a multicast ATM switch

Bibliography

[1] Thomas R Bannıza, Gert J Eılenberger, Bart Pauwels, and Yves Therasse, "Design and Technology Aspects of VLSI's for ATM Switches," *IEEE J. Select Areas Commun* , vol 9, no 8 pp 1255-1264, Oct 1991

[2] Jae W Byun and Tony T Lee, "The Design and Analysis of an ATM Multicast Switch with Adaptive Traffic Controller," *IEEE/ACM Trans Networking*, vol. 2, no 3, pp 288-298, June 1994

[3] Tom Chaney, J Andrew Fingerhut, Margaret Flucke, Jonathan S Turner, "Design of a Gigabit ATM Switch ," *WUCS-96-07*, URL http //www cs wustl edu/cs/tech-reports/wucs-96-07 ps Z, Washington University, St Louis

[4] H Jonathan Chao, "A Recursive Modular Terabit/Second ATM Switch ," *IEEE J Select Areas Commun* , vol 9, no 8, pp 1161-1172, October 1991

[5] H Jonathan Chao and Byeong-Seog Choe, "Design and Analysis of a Large-Scale Multicast Output Buffered ATM Switch ," *IEEE/ACM Trans Networking*, vol. 3, no. 2, pp 126-138, April 1995

[6] David X Chen and Jon W Mark, "SCOQ A Fast Packet Switch with Shared Concentration and Output Queueing ," *Proc of IEEE INFOCOM*, pp 145-154, 1991

[7] R L Cruz, "The statistical data fork A class of broadband multichannel switches," *IEEE Trans Commun* , pp 1625-1634 October 1992

[8] Kai Y Eng, M J Karol and Yu-Shuan Yeh, "A Growable Packet (ATM) Switch Architecture Design Principles and Applications," *IEEE Trans Commun* , vol. 40 no 2, pp 423-430, Feb 1992

[9] A P J Engbersen, "IBM's ATM switching technology," URL · http //www zurich ibm com/Technology/ATM/SWOCPWP/

[10] Noboru Endo, Takahito Kozaki Toshiya Ohuchi, Hiroshi Kuwahara and Shinobu Gohara, "Shared Buffer Memory Switch for an ATM Exchange." *IEEE Trans Commun* , vol 41, no 1, pp 237-145, Jan 1993

[11] ForeRunner ATM Switch Architecture, "ATM White Paper," *Fore Systems*, April 1996

[12] Paulo Giacomazzi, "Channel Grouping Techniques in the Tandem Banyan Switching Fabric," *Proc of GLOBECOM*, pp 302-308, 1994

[13] J N Giacopelli, J J Hickey, W S Marcus, W D Sincoskie and M Littlewood, "Sunshine A high-performance self-routing broadband packet switch architecture," *IEEE J Select Areas Commun* , vol 9, pp 1289-1298, October 1991.

[14] S Gupta, "An improved multicast scheme and VLSI analysis of a new layered ATM switch," *M Tech Thesis*, IIT Kanpur (India), 1996

[15] Rainer Handel, Manfred N Huber, and Stefan Schroder, "ATM Networks - Concepts, Protocols, Applications," *Addison-Wesley Publishing Company*, Second edition

[16] Michael G Hluchyj, M J Karol, "Queueing in High Performance Packet Switching," *IEEE J Select Areas Commun* , vol 6, no 9, pp 1587-1597 December 1988

[17] J Y Hui, "Switching and traffic control for integrated broadband networks"

[18] "Technical Reference, Lightstream 1010," *Cisco Systems*

[19] Farouk Kamoun, Leonard Kleinrock, "Analysis of Shared Finite Storage in a Computer Network Node Environment under General Traffic Conditions," *IEEE Trans Commun* , vol comm-28, no 7, pp 992-1003, July 1988

[20] Rajgopal Kannan, Radim Bartoš, Kyungsook Y Lee, and Harry F. Jordan, "SXmin A Self-Routing, High-Performance ATM Packet Switch Based on Group-Knockout Principle," *Proc of GLOBECOM*, pp 453-457, 1994

[21] Mark J Karol, "Buffer management in a packet switch," *IEEE Trans. Commun*, vol COM-26 no 3, pp 328-337, March 1978

[22] M J Karol, M G Hluchyj and S P Morgan, "Input versus output queueing on a space-division packet switch ' *IEEE Trans Commun*, COM-35 pp 1347-1356. December 1987

[23] M J Karol and Chih-Lin I, "A Performance Analysis of a Growable Architecture for Broad-Band Packet (ATM) Switching," *IEEE Trans Commun*, vol. 40 no 2 pp 431-439, Feb 1992

[24] Takahiko Kozaki, Noboru Endo, Yoshito Sakurai, Osamu Matsurbara, Masao Mizukami, and Ken'ichi Asano, "32 × 32 Shared Buffer Type ATM Switch VLSI s for B-ISDN's," *IEEE J Select Areas Commun*, vol 9, no 8, pp 1239-1247, Oct 1991

[25] Taek-Geun Kwon, Choong-Kyo Jeong, "A Simple, Extendible ATM Switch with load-balanced Rounding Copy Network," *ICC-95*, pp 1122-1126

[26] Chin-Tau Lea, "A multicast broadband packet switch," *IEEE Trans Commun*, vol. 41, no 4, pp 621-630, April 1993

[27] T T Lee, "Nonblocking copy networks for multicast packet switching," *IEEE J Select Areas Commun*, vol 6, no 9, pp 1455-1467, December 1988.

[28] Tony T Lee, "A Modular Architecture for Very Large Packet Switches ," *IEEE Trans. Commun*, vol 38, no 7, pp 1097-1106, July 1990

[29] Soung C Liew, "A General Packet Replication Scheme for Multicasting with Application to Shuffle-Exchange Networks," *IEEE Trans Commun*, vol 44, no 8, pp 1021-1033, August 1990

[30] Soung C Liew, "A General Packet Replication Scheme for Multicasting in Interconnection Networks " *Proc of INFOCOM*, pp 394-400, 1995

[31] Xinyi Liu and H T Mouftah, "A Dynamic Cell_Splitting Copy Network Design for Multicast Switching," *Proc of GLOBECOM*, pp 458-462, 1994.

[32] Willium S Marcus, "A CMOS Batcher and Banyan Chip Set for B-ISDN Packet SWitching," *IEEE J Solid State Circuits*, vol 25, no 6, pp 1426-1432, December 1990

[33] P S Min, H Saidi and M V Hegde, "A nonblocking architecture for broadband multichannel switching " *IEEE/ACM Trans Networking*, pp 181-198, April 1995

[34] Achille Pattavina, "Nonblocking Architectures for ATM Switching," *IEEE Communications Magazine*, pp 38-48, Feb 1993

[35] Achille Pattavina, "Multichannel Bandwidth Allocation in a Broadband Packet Switch," *IEEE J Select Areas Commun* , vol 6, no 9, pp 1489-1499, December 1988

[36] Jayesh V Shah and D Manjunath, "Memory Optimized Lee Multicast Switch," *Proc of NCC*, 1997

[37] Yasuro Shobatake, Masahiko Motoyama, Emiko Shobatake, Takashi Kamitake, Shoichi Shimizu, Makoto Noda, and Kenji Sakaoue, "A One-Chip Scalable 8*8 ATM Switch LSI Employing Shared Buffer Architecture," *IEEE J Select Areas Commun* , vol 9, no 8, pp 1248-1253, Oct 1991

[38] Sandeep Sibal and Ji Zhang, "On a Class of Banyan Networks and Tandem Banyan Switching Fabrics," *Proc of IEEE INFOCOMM*, pp 481-488, 1993

[39] Robert J Simcoe and Tong-Bi Pei, "Perspectives of ATM switch architecture and the influence of traffic pattern assumptions on switch design," *Proceedings of ACM SIGComm*, pp 93-105, 1995

[40] C L. Tarng and J S Meditch, "A high performance copy network for B-ISDN," *Proc of IEEE INFOCOM*, pp 171-180, 1991

[41] Fouad A Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks," *Proceedings of the IEEE*, vol 78, no 1, pp 133-166, Jan 1990

[42] Fouad A Tobagi, Timothy Kwok, and Fabio M Chiussi "Architecture, Performance, and Implementation of the Tandem Banyan Fast Packet Switch," *IEEE J Select Areas Commun.*, vol 9, no 8, pp 1173-1193, October 1991

[43] J S Turner, "Design of a broadcast packet switching network," *IEEE Trans Commun* , vol 36, no 6, pp 734-743, June 1988

[44] J S Turner, "A practical version of Lee's Multicast Switch architecture," *IEEE Trans Commun* , vol 41, no 8, pp 1166-1169, August 1993

[45] J S Turner, "An optimal nonblocking multicast virtual channel switch," *Proc of IEEE INFOCOM*, pp 298-305, 1994

[46] Y S Yeh, M G Hluchyj and A S Acampora, "The Knockout Switch A simple architecture for high-performance packet switching," *IEEE J Select Areas Commun* , vol SAC-5, pp 1274-1283, October 1987

[47] Ellen E Witte, "A Quantitative Comparison of Architectures for ATM Switching Systems ," *WUCS-91-47*, Washington University, St Louis

[48] Ellen Witte Zegura, "Architectures for ATM Switching Systems," *IEEE Communications Magazine*, pp 28-37, Feb 1993

[49] Wen De Zhong, Yoshikuni Onozato, and Jaidev Kaniyil, "A Copy Network with Shared Buffers for Large-Scale Multicast ATM Switching," *IEEE/ACM Trans Networking*, vol 1, no 2, pp 157-165, April 1993

EE-1997-M-SHA-ON